目录

PART 1：原理与技术

第一章 简介

第二章 数学基础

第三章 采样与量化

第四章 变换法

第八章 图像分割

第九章 表示与描述

第十章 图像模式识别

PART 2：Matlab 实习

第一章 Matlab 实验常用函数简介

第二章 实验——数学基础


PART 1
原理與技術



第一章

简　介



百闻不如一见

1.1 背 景

我们常说百闻不如一见，这是因为人的视觉、感观对图像非常强烈，往往用笔墨还不足以形容。例如，陈述一个小女孩有多可爱，与其用尽所有形容词，不如出示一张她的照片。

显然图像是传达信息非常直接又有效的方式，但是倘若上述照片没有拍好（例如光线不足及拍摄时手部颤抖等问题），则其信息传达的效果可能还不如用言语形容。虽然有些照片可以重拍，但也有很多情况是很难或甚至无法再重拍的。例如负责太空探测的太空船所传回的珍贵照片，由于经济及时效上的考虑，通常不会有第二次重拍的机会。在我们日常生活当中也有许多珍贵的历史性镜头是不能有重拍机会的。

对于效果不佳的图片，特别是记录珍贵历史性时刻者，如果能以经济又有效的方式加以处理，使其达到原本能传递的信息量，是不是很好呢？这就是*图像处理* (image processing) 所要探讨的课题，换言之就是如何*有效* (effective) 又*有效率* (efficient) 地来处理图像，使其能传达我们所需要的信息。

那*数字图像处理* (digital image processing) 又是什么呢？简言之就是将所撷取到的图像*数字化* (digitize)，以利电脑的处理。比起像摄影师暗房技巧那一类的所谓光学图像处理以及像调整电视机旋钮改变电压使对比度或亮度改变的类似图像处理，数字图像处理有更大的*弹性* (flexibility) 以及效率。数字图像处理除可改善图像信息使人理解外，它还能使机器像人一样具有视觉感官的能力。不论是过去许多已成功的数字图像处理应用还是更多正在开发中的应用，都属这两类。

虽然图像处理技术最早期的应用之一可回溯到 20 世纪初，用来改善伦敦和纽约间经海底电缆发送的图像的品质，但一直到 20 世纪 50 年代，随着大型数字计算机和太空科学研究计划的出现，大家才注意到图像处理的潜力。1964 年在美国航太总署的喷射推进实验室开始用计算机技术改善从太空探测器获得的图像，当时用计算机处理由*巡航者七号* (Ranger 7) 传回的月球图片，以校正电视摄影机所存在的几何失真或响应失真。其后有一连串的星际探测计划，一直到现在都在持续不断地送回更多图像。

从 1964 年迄今，图像处理领域一直快速地发展着。除了在太空计划中的应用，目前数字图像处理技术还用于解决其他问题。主要的应用领域如下：

- **生物医学领域**

首先用于细胞分类、染色体分类和放射图像的处理。1972 年 X 光*电脑断层扫描* (computer tomography, CT) 获得实际应用，在医学工程上为一大突破；1977 年白血球自动分析仪问世；1980 年研究出心脏活动立体图像的技术。医学图像的种类包括 X 光图像、同位素图像、核磁共振图像、超声波图像、红外线图像以及显微图像等。对这些图像作对比度增强或伪彩色等的处理可帮助医师诊断如肺病、肿瘤及心血管等疾病。

- **遥测数据分析**

遥测常以颜色为依据，由飞机、卫星及太空船上的*多光谱* (multi-spectral) 扫描器摄得图

像，范围从可见光至红外光 (有时含紫外光) 分成几个频带，每个频带所摄得的颜色都不同，可用于土地使用、作物收成、作物病害侦测、森林及水资源调查、环境污染侦测、地质与地形分析、矿藏探勘及气象预测等。由于数据量庞大，因此寻求处理及分析这些图像的自动方法，特别是图像对比度增强、分割及图像识别的技术显得极为重要。

- **科学研究**

例如考古学可用图像处理方法恢复模糊或其他降质状况的珍贵文物图像。这些文物在拍成照片后已遗失或损坏，因此这些照片成为唯一可用的记录。又如在物理学上可增强在高能态电浆及电子显微镜等场合所拍下的实验图像，以增加对实验结果的了解。

- **一般工商业应用**

这一方面已有许多实例，例如用无损图像处理方式检查零件内部的瑕疵及焊接品质、金属材料的成分与结构分析、纺织品品质检测、印刷电路板及焊点不良检查、零件安装检查等。又如邮政编码自动读取系统、运动员运动图像分析、机器人自走、道路状况判断、组装零件识别、光学字的读取与辨认、条形码读取、指纹、瞳孔、颜面等图像的身份识别、试装及发型设计等。

- **通讯与电脑资料存储**

包括传真机、可视电话及电视会议等都采用图像数据压缩的技术，使通讯更快速。此外，采用同样技术可大大降低图像数据存储量以降低存储负担。

整个图像处理的领域仍在蓬勃发展中，其原因有：① 电脑功能对价格比越来越高；② 图像撷取与显示设备更加普遍与便利；③ 图像处理的观念普及，容易创造新的应用。预期数字图像处理在我们未来生活中将扮演愈来愈重要的角色。

1.2　图像的表示

图像最广义的观点是指视觉信息。举凡照片、图画、电视画面以及由透镜、光栅及全息图 (hologram) 所构成的光学成像等均属之。

令 $F(x,y,t,\lambda)$ 代表上述图像源在空间坐标 (x, y)，时间 t 且波长为 λ 情况下的空间能量分布，此分布通常可视为一个具正实数且有上界的光强度函数，亦即

$$0 < F(x,y,t,\lambda) \leq A \tag{1.2-1}$$

其中 A 为最大的图像强度。此外不论就实际图像的考虑或数学上讨论的方便，x, y 与 t 应该也都是有界的，例如

$$\begin{aligned} &0 \leq x \leq L_x \\ &0 \leq y \leq L_y \\ &0 \leq t \leq T \end{aligned} \tag{1.2-2}$$

我们对上述光强度函数的反应常用瞬间强度来表示

$$f(x,y,t)=\int_0^\infty F(x,y,t,\lambda)V(\lambda)\mathrm{d}\lambda \tag{1.2-3}$$

其中$V(\lambda)$为相对亮度效率函数，换言之是人类视觉的频谱响应。同理，人对色彩的反应也可用类似 (1.2-3) 式的方式来表示。对一个任意的红—绿—蓝坐标系统，其三基色的瞬间值为

$$f_R(x,y,t)=\int_0^\infty F(x,y,t,\lambda)R(\lambda)\mathrm{d}\lambda \tag{1.2-4a}$$

$$f_G(x,y,t)=\int_0^\infty F(x,y,t,\lambda)G(\lambda)\mathrm{d}\lambda \tag{1.2-4b}$$

$$f_B(x,y,t)=\int_0^\infty F(x,y,t,\lambda)B(\lambda)\mathrm{d}\lambda \tag{1.2-4c}$$

其中$R(\lambda),G(\lambda),B(\lambda)$分别为对红、绿、蓝三基色的频谱响应。如果是更多个感应的图像，则第 i 个频谱的图像为

$$f_i(x,y,t)=\int_0^\infty F(x,y,t,\lambda)S_i(\lambda)\mathrm{d}\lambda \tag{1.2-5}$$

其中$S_i(\lambda)$是第 i 个感应器的频谱响应。

如果图像内容不随时间变化，或相当于$t=t_0$ (一个定值) 时所得图像，则 (1.2-1) 式到 (1.2-5) 式中的时间因子可去掉，此时所得图像称为静止图像 (例如一张照片)，反之称为活动图像 (例如电视的连续画面)。

因此一个单色 (monochrome) 静止图像可用一个二维的光强度函数$f(x,y)$来表示，其中 x 与 y 表示空间坐标，而在任意点(x,y)的 f 值与在该点图像的亮度 (或灰度) 成正比。一个数字图像是图像$f(x,y)$在空间坐标和亮度上都数字化后的图像。可将数字图像视为一个矩阵，矩阵行与列的值决定一个点，而对应的矩阵元素值就是该点的灰度。这种矩阵的元素称为像素 (picture element 或 pixel)，所对应的灰度可称为像素值。因此像素可以说是数字图像的最基本单位。

1.3　数字图像处理

数字图像处理就是利用电脑对数字图像做运算，以达到 1.1 节所述的许多应用。显然要有效解决众多的图像处理应用问题，有时必须研究出专门为其量身打造的图像处理方法，不过大致上可将这些问题及其图像处理方式归纳成以下几类，这也是本书主要探讨的范围。

- **图像增强**

图像增强 (enhancement) 是用来强调图像的某些特征，以便于作进一步分析或显示。例如对比度的增强是用来使对比度低的图像更容易显现其特征，而低对比度的可能原因包括光线不足、图像感应器的动态范围不够以及在图像摄取时光圈设定错误等。图像增强的过程本身并没有增加原资料所含的信息，它只是把图像某些部分的特性更加强调罢了。图像增强的算

法通常是交互式 (interactive) 而且与所考虑的应用有密切的关系。图 1.3-1 是一个图像增强的实例，其中 (a) 图是一低对比度图像，(b) 图则是其经过图像增强处理后的结果。

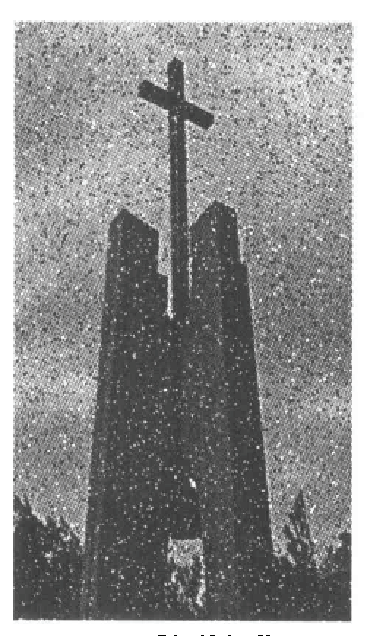

(a) 低对比度图像


(b) 经图像增强处理后的图像

图 1.3-1 图像增强实例

● **图像恢复**

图像恢复 (restoration) 是指将图像已知的降质因素去掉或减少。使图像降质的因素包括感应器或拍摄环境产生的干扰、没有对焦所造成的模糊、摄像机与物体间相对运动所造成的模糊、感应器的非线性几何失真等。图像恢复的目的是试图将受污染或降质的图像带回到原本不受污染的状况之下所应得的干净图像。虽然它与图像增强都会造成视觉上较佳的感受，但后者比较关切的是图像特征增强或抽取而不是去除退化或污染。图 1.3-2 是一个图像恢复的实例，其中图 (a) 是受干扰污染的图像，(b) 图则是其经过图像恢复后的结果。


(a) 受污染图像

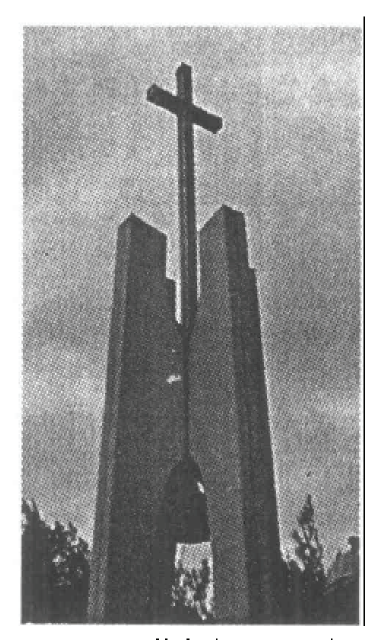
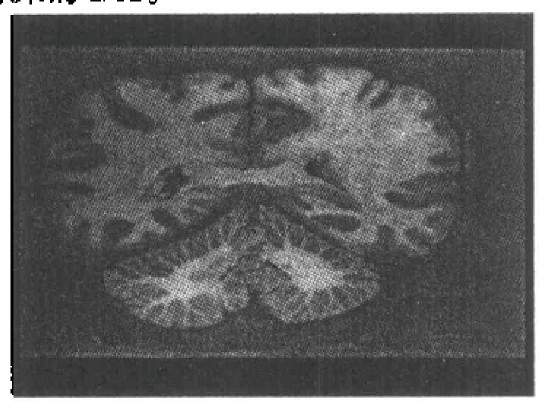

(b) 经恢复处理的图像

图 1.3-2 图像恢复实例

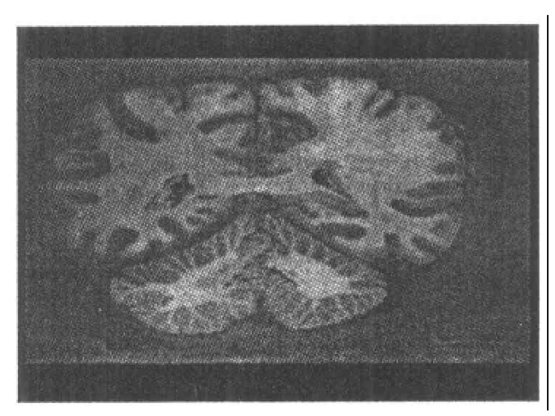
● **图像压缩**

图像压缩 (compression) 的目的是在减低代表数字图像所需的数据量，这样做的好处是可以减少图像传输时间及储存空间。许多人都有使用网络浏览器的经验，如果是单纯的文字资

料，通常传输速度较快，等待时间也较短；但如果夹杂图像，则等待时间较长，而且等待时间大约与图像大小成正比。这是为什么有人倡议在网络堵塞的情况下，WWW 的首页应尽量减少不必要的图像，或用较小的图像。图像数据量到底有多大？以一个 1 024×1 024 大小且每个像素 8 bits (1 byte) 的单色图像为例，约是 1MB 的存储量。若以 38.4 kbps 的速率传送则需 1 024×1 024×8 / 38 400 (秒)或约为 3 分 38 秒才传送完毕。图 1.3-3 是一个图像压缩的实例，其中 (a) 图是原始图像，(b) 图是经 12 倍压缩恢复重建后的图像，其中 12 倍压缩是指数据量仅为原来的 1/12。



(a) 原始图像



(b) 经 12 倍压缩恢复的图像

图 1.3-3　图像压缩实例

- **图像重建**

此图像重建 (reconstruction) 的工作是由几个一维的图像投影来重建出更高维的物体图像。一个图像投影的取得是以平行的 X 光 (或其他放射穿透) 光束照射物体并在物体背面接收此投影量，接着在同一平面上改变光束照射的角度以获得不同的投影，再以某些重建算法将这些投影组合或此物体的一个横剖面图像。此种技术主要用于医学图像 (电脑断层扫描)、天文学星象观测、雷达图像处理、地质研究及无损检测等。

- **图像分析**

图像分析 (analysis) 是试图从图像中提取并描述某些特征，以自动产生所需信息。其目标是赋予计算机或机器像人一样看东西的能力。例如要像人一样阅读一本书，至少需具备识别字元及了解文意的功能，更不用说归纳、整理及推论等能力了。由此可知要做图像分析，必须使机器具备某种程度的智慧 (intelligence)。一些智慧的特征包括：① 能从含有许多不相干细节的背景中找到所要的信息；② 能从范例中学习并将所学知识应用推广到其他状况中；③ 能从不完整的资料中推论出完整的资料。

1.4　数字图像处理系统

一个基本的数字图像处理系统应包括计算机处理或存储单元，以及显像或记录单元。如果要处理自己撷取的图像，则必须再加人撷取单元，整个基本结构如图 1.4-1 所示。

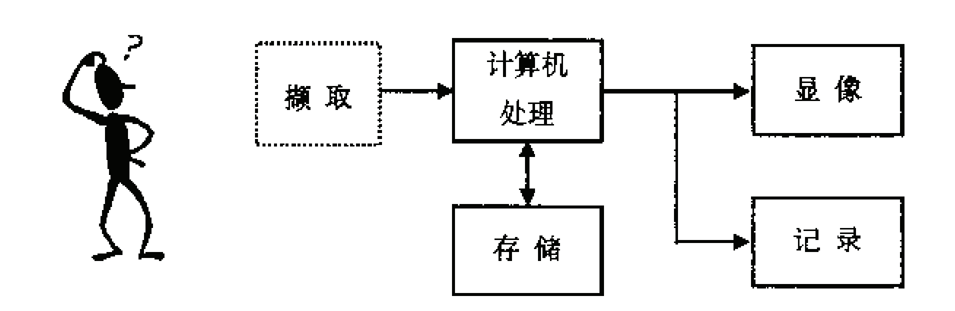


图 1.4-1 数字图像处理的基本结构

1.4-1 撷 取

撷取的目的在于获取一个数字图像。其所用的装置统称为数字化仪 (digitizer)。

数字化仪含下列四大组成元素：

① 取样孔隙 (sampling aperture)：允许数字化仪个别获取每一个像素，而暂忽略图像的其他部分。

② 图像扫描装置：将取样孔隙依一定模式在图像上移动、使像素按某种顺序排列。

③ 光感应器 (sensor)：可度量光透过取样孔隙所获得的图像的亮度。此感应器基本上是一种将光的强度转变成电压或电流的转换器。

④ 量化器 (quantizer)：将感应器连续性数值的输出变成有限个可能值。通常这是由模拟—数字 (analog-to-digital)转换器的电子电路所达成。

常用的光源有白炽灯 (incandescent bulb)、激光、发光性合成物质 (phosphor) 以及发光二极管 (light-emitting diode)。常用的光感应器有光放射元件 (photoemissive device)、硅感应器、光二极管及光电晶体等。扫描装置分机械式扫描元件、电子束扫描、电场偏转及磁场偏转扫描等。

最近研制出的图像感应器是电子式的自我扫描固态感应器阵列。主要有三种：分别是电荷耦合元件 (charge couple device, CCD) 阵列、电荷注入元件 (charge injection device, CID) 阵列及光二极管阵列。所有这些元件都在一个单一的集成电路晶片上有一个线性或矩形的光感应阵列，而且都有将入射图像所产生的载子电荷读出所需要的电子电路。

1.4-2 储 存

一个大小 1 024×1 024 的 12 位图像需要1.5 MB (1.5×10^6 bytes) 的储存空间，所以需求相当高。有人估计在一个有 1 500 张病床的大学医院的数字图像资料，每年就要超过 20 TB (20×10^{12} bytes) 的储存空间。

按储存时间长短，储存方式大致可分为三类：① 处理过程所需要的短暂储存；② 经常需要存取的储存；③ 较少存取的存档存储 (archival storage)。

短暂储存的简单例子就是电脑中极为重要的存储单元之一——RAM，即随机存储单元。

另一个例子就是一个称为显示缓冲区 (frame buffer) 的板子，其特色是可快速存取，通常以每秒 30 个画面的速度做更新。

经常性存取储存的例子就是磁盘，包括软盘及硬盘。最近发展的磁光盘 (magneto-optical, MO) 储存，利用激光及特殊材料的技术，在 5 (1/4) 吋光盘上可有将近 1 GB (1×10^9 bytes) 的容量。

存档存储的两个特色是超大容量的需求和存取较不频繁。最佳的选择是写一次读多次 (write-once-read-many, WORM) 的光盘，目前已有 10 GB 以上容量的光盘。

1.4-3 计算机处理

数字图像处理常涉及大量的运算，因此需要计算机高速计算能力的帮助。除了有即时性的速度要求才需要特殊硬件计算外，一般都是在通用型计算机上以软件将图像处理的算法实现而达到处理的效果。市面上已有许多专为图像处理所设计的套装软件，具备一般图像处理的功能，此类软件通常都配有很友好的图形介面 (graphics user interface, GUI)，相当容易使用。

此外，还可用一般用途的编程语言来实现图像处理的工作，例如 C、C++、Matlab 与 Java 等。本书所附程序范例就是以 Matlab 的语言写成的。

1.4-4 显　示

图像处理前后的结果通常可借助显示设备呈现。单色与彩色电视监视器是目前最常用的显示设备。通常衡量一个显示设备需考虑下列因素：

① 显示的图像的大小：一个考虑是实际显示设备的大小，一个是可容许显示的图像的大小。

② 亮度 (photometric)解析度：这是指在每个像素位置产生正确亮度的准确性。例如能产生多少个不同的灰度级：4 位存储能处理 16 个灰度级，8 位存储能处理 256 个灰度级等等。要注意若显示设备有干扰，则号称能处理 256 个灰度级的显示设备未必能稳定地真正呈现出 256 个不同的灰度级。

③ 灰度的线性：这是指亮度与输入灰度之间成比例的程度。任何显示设备都有一个输入灰度级对输出亮度的转换函数，此条函数曲线的理想状况是一条直线，或是有明显的一段为直线。

④ 显示校正：一般常用一个含有各种不同灰度级的长条图、方块、图形等几何图案所构成的标准图像来当作校正的基准。适当的校正可使整个亮度范围得以正常显示。

⑤ 低频响应：这是衡量显示系统对显示大面积相同灰度的能力，这个因素与显示点的样式、显示点之间的距离以及干扰大小与位置有关。

⑥ 高频响应：这可用来表示显示系统显示精致细节部分的能力。

1.4-5　记　录

这里所谓的记录是指以打印或其他“永久”性呈现图像的方式，有别于先前由显示设备所呈现的图像在电源关闭后就消失的状况。

将图像永久记录在纸上或软盘上的装置称为图像记录器。此记录器一般采用下列技术：

① 抖动 (dithering)：有些打印技术能在每个像素上打印出所有可能的灰度，有些则只能打印黑点或白点 (即不打印的点)。后者可借助称为抖动或半色调 (halftone) 的程序来模拟出不同灰度的效果，其概念是设计出大小甚至样式均不相同的黑点，然后对于较光亮的画面就用较小的点，较黑的部分则用较大的点。在一定距离以外观看其打印结果，可明显感受到不同灰度的效果。

② 彩色打印：重建图像所用的是三种色料，分别是吸收红光的蓝绿色 (cyan)、吸收绿光的紫红色 (magenta)，以及吸收蓝光的黄色，此三种颜色简称为 CMY 系统。理论上等量的蓝绿色、紫红色与黄色可混合成黑色，但是实际上因为只能吸收部分入射光线，故呈现灰色，此种混成的黑色称为合成黑 (composite black)。为了改善此现象，通常采用全黑的第四种颜料而形成所谓的 CMYK 系统。

1.5　本书结构

本书共分两大部分，第一部分介绍数字图像处理的原理与技术，第二部分则是与其搭配的实习教材。第一部分共计十章，其中前三章是背景基础的介绍，第四到第七章是典型的图像处理工作，最后三章则与图像分析有关。以下略加介绍：

第一章　简介：介绍数字图像处理的基本概念。

第二章　数学基础：提供数字图像处理常用到的数学基础。

第三章　取样与量化：图像数字化的原理是本章中的重点。

第四章　变换法：介绍基本变换的原理以及各种常见的图像变换。

第五章　图像增强：探讨图像增强的技术。

第六章　图像恢复：说明图像恢复的原理与方法。

第七章　图像压缩：介绍编码原理及常用的图像压缩方法。

第八章　图像分割：讨论图像如何分割并列举各种图像分割的方法。

第九章　表示与描述：图像如何有效描述以便进一步加以辨认是本章探讨重点。

第十章　图像模式识别：介绍图像模式识别原理以及各类辨识的方法。

除了第一章外，其余每章均在第二部分配有以 Matlab 语言写成的实习教材，包括原理、程序范例、结果展示、讨论以及动手做等单元供读者参考使用。所有相关的程序范例以及图像文件都放在所附的光盘中。

1.6 主要参考文献

数字图像处理历经全世界学者大量研究的结果，其著作相当丰富，各种专著或期刊、杂志上发表的文章不计其数。此外，每年都有许多相关的学术研讨会议，例如 IEEE International Conference on Image Processing，在国内较知名且历史较久的则如 CVGIP (Computer Vision, Graphics, and Image Processing)会议，这是由台湾地区图像处理与图形识别学会所主办的会议。

为方便读者查阅，以下列出与数字图像处理有关的主要书籍与期刊。

书 籍

Rafael C. Gonzalez and Richard E. Wood, Digital Image Processing, Addison-Wesley, 1992.

William K. Pratt, Digital Image Processing, 2nd ed., John Wiley & Sons, 1991.

Kenneth R. Castleman, Digital Image Processing, Prentice-Hall, 1996.

Anil K. Jain, Fundamentals of Digital Image Processing, Prentice-Hall, 1989.

Jae S. Lim, Two-Dimensional Signal and Image Processing, Prentice-Hall, 1990.

R. Rosenfeld and A. C. Kak, Digital Picture Processing, 2nd ed., vols. 1 & 2, Academic Press, 1982.

期 刊

Applied Optics

Computers in Biology and Medicine

Computer Vision, Graphics, and Image Processing

Engineering in Medicine and Biology Magazine

Electronics Letters

IEE Proceeding - Vision Image and Signal Processing

IEEE Transactions on Image Processing

IEEE Transactions on Pattern Analysis and Machine Intelligence

IEEE Transactions on Signal Processing

IEEE Transactions on Communications

IEEE Transactions on Medical Imaging

IEEE Transactions on Circuits and Systems, Part Ⅱ: Analog and Digital Signal Processing

IEEE Transactions on Circuits and Systems for Video Technology

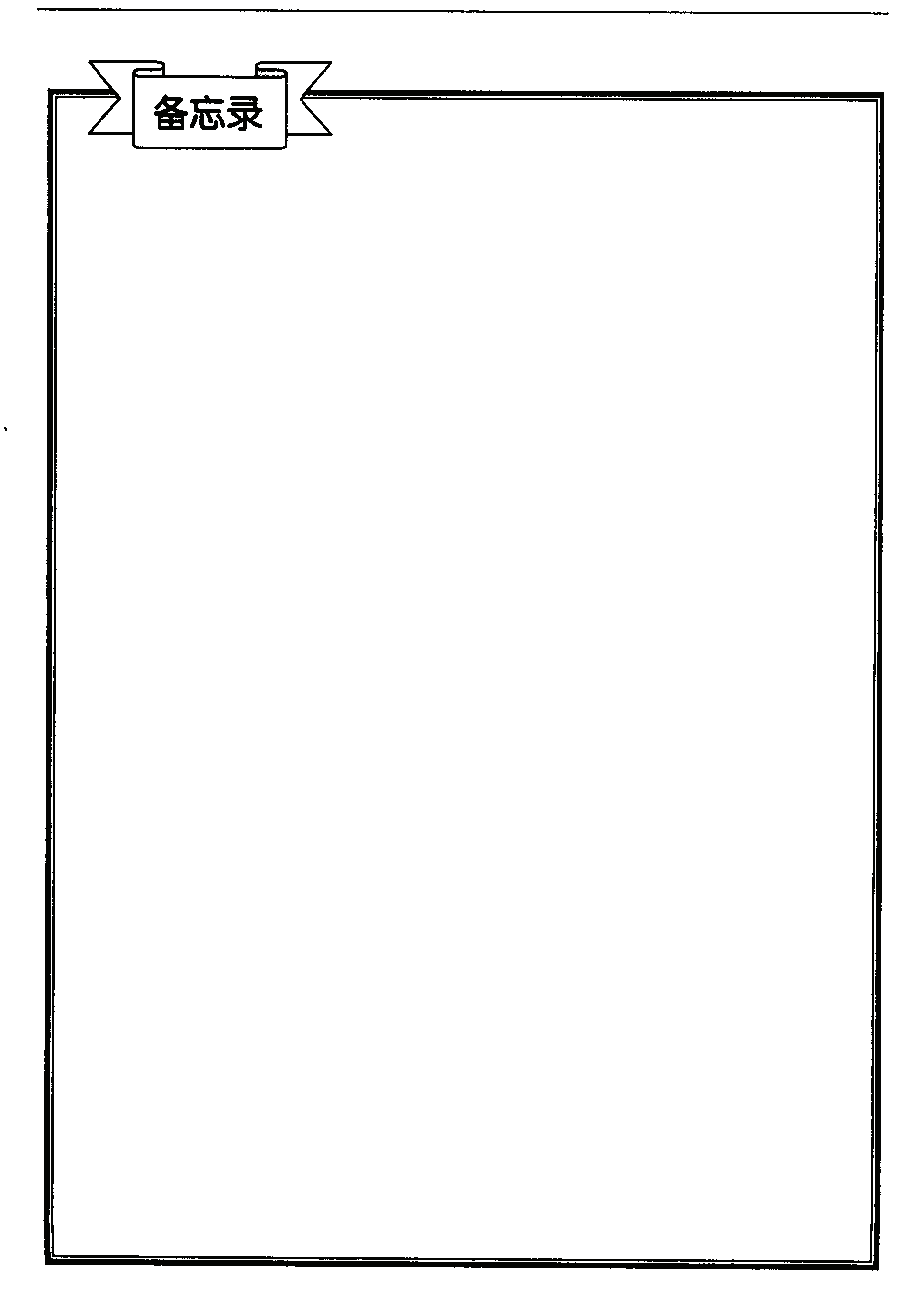
IEEE Transactions on Multimedia

IEEE Transactions on Biomedical Engineering

IEEE Transactions on Information Technology in Biomedicine

IEEE Transactions on Information Theory

IEEE Transactions on System, Man, and Cybernetics
Medical and Biological Engineering and Computing
Optical Engineering
Pattern Recognition
Pattern Recognition Letters
Proceedings of the IEEE
Signal Processing
Signal Processing Magazine
Signal Processing Letters
Signal Processing: Image Communication
Ultrasonic·Imaging


备忘录

第二章

数学基础



16 217 217 …
14 215 …
14 214
10
⋮ ⋮
78 86
140 112 80 92
178 164 141 100 96
… 180 173 164 133 119
173 175 174 172 154 140

本章将讨论在数字图像处理技术中较常用到的一些数学基础。由于图像通常由二维系统来做描述，因此我们从二维的信号与系统讨论起。

2.1 信　号

此处我们所考虑的信号为离散空间 (discrete-space) 信号。一个二维 (2-D) 离散空间信号 (亦称序列 (sequence)) 为一个具有两个整数变量的函数。例如，$x(n_1,n_2)$ 表示一个定义为 n_1 与 n_2 所有整数值的序列。需注意的是所有 n_1 与 n_2 的非整数值所造成的 $x(n_1,n_2)$ 函数值并非为零，而是没有定义。

某些特定序列在二维信号处理中扮演重要的角色，这些序列有脉冲 (impulse) 序列、单位阶跃 (step) 序列、指数 (exponential) 序列、可分离 (separable) 序列及周期 (periodic) 序列。

● **脉冲序列**

脉冲（或称单位取样）序列，以 $\delta(n_1,n_2)$ 表示，定义为

$$\delta(n_1,n_2)=\begin{cases}1, & n_1=n_2=0\\ 0, & 其他情况\end{cases} \tag{2.1-1}$$

一个二维序列 $\delta(n_1,n_2)$，如图 2.1-1 所示，它与一维信号处理中的脉冲 $\delta(n_1,n_2)$ 扮演相同的角色。

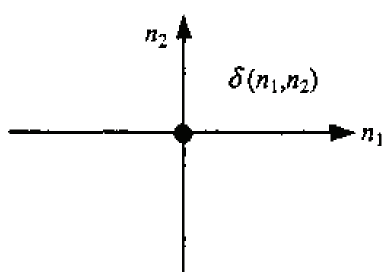


图 2.1-1　脉冲序列 $\delta(n_1,n_2)$

任何序列 $x(n_1,n_2)$ 都可以被表示成时延脉冲的线性组合：

$$\begin{aligned}x(n_1,\ n_2)=&\cdots+x(-1,\ -1)\delta(n_1+1,\ n_2+1)+x(0,\ -1)\delta(n_1,\ n_2+1)\\&+x(1,\ -1)\delta(n_1-1,\ n_2+1)+\cdots+x(-1,\ 0)\delta(n_1+1,\ n_2)\\&+x(0,\ 0)\delta(n_1,\ n_2)+x(1,\ 0)\delta(n_1-1,\ n_2)+\cdots\\&+x(-1,\ 1)\delta(n_1+1,\ n_2-1)+x(0,\ 1)\delta(n_1,\ n_2-1)\\&+x(1,\ 1)\delta(n_1-1,\ n_2-1)+\cdots\\=&\sum_{k_1=-\infty}^{\infty}\sum_{k_2=-\infty}^{\infty}x(k_1,\ k_2)\delta(n_1-k_1,\ n_2-k_2)\end{aligned} \tag{2.1-2}$$

在系统分析中 $x(n_1,n_2)$ 的表示式将会非常有用。

线脉冲构成一种在一维信号中不存在的序列类别；一个线脉冲的实例为如图 2.1-2 所示

的二维序列$\delta_T(n_1)$，其定义为

$$x(n_1,n_2)=\delta_T(n_1)=\begin{cases}1, & n_1=0\\ 0, & \text{其他情况}\end{cases} \tag{2.1-3}$$

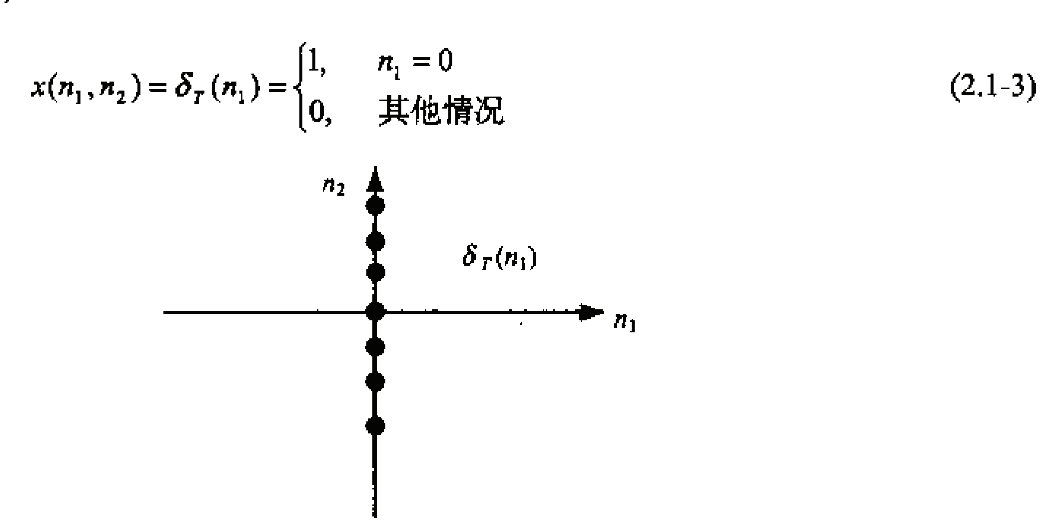


图 2.1-2　线脉冲$\delta_T(n_1)$

其他包括$\delta_T(n_2)$及$\delta_T(n_1-n_2)$在内的例子，其定义均与$\delta_T(n_1)$类似。$\delta_T(n_1)$中的下标T指出$\delta_T(n_1)$为一个二维序列，这个符号是用来避免当二维序列为一单一变量的函数时产生混淆，例如：如果没有下标T，$\delta_T(n_1)$可能会与一维脉冲$\delta(n_1)$混淆。

- **单位阶跃序列**

单位阶跃序列，表示为$u(n_1,n_2)$，其定义为

$$u(n_1,n_2)=\begin{cases}1, & n_1,n_2\geqslant 0\\ 0, & \text{其他情况}\end{cases} \tag{2.1-4}$$

序列$u(n_1,n_2)$如图 2.1-3 所示，它与$\delta(n_1,n_2)$有下列的关系

$$u(n_1,n_2)=\sum_{k_1=-\infty}^{n_1}\sum_{k_2=-\infty}^{n_2}\delta(k_1,k_2) \tag{2.1-5}$$

或者

$$\delta(n_1,n_2)=u(n_1,n_2)-u(n_1-1,n_2)-u(n_1,n_2-1)+u(n_1-1,n_2-1) \tag{2.1-6}$$

部分单位阶跃序列在一维空间中并没有类似的部分。如图 2.1-4 所示为二维序列的例子$u_T(n_1)$，其定义为

$$x(n_1,n_2)=u_T(n_1)=\begin{cases}1, & n_1\geqslant 0\\ 0, & \text{其他情况}\end{cases} \tag{2.1-7}$$

其他例子包括$u_T(n_2)$与$u_T(n_1-n_2)$，其定义均与$u_T(n_1)$相似。

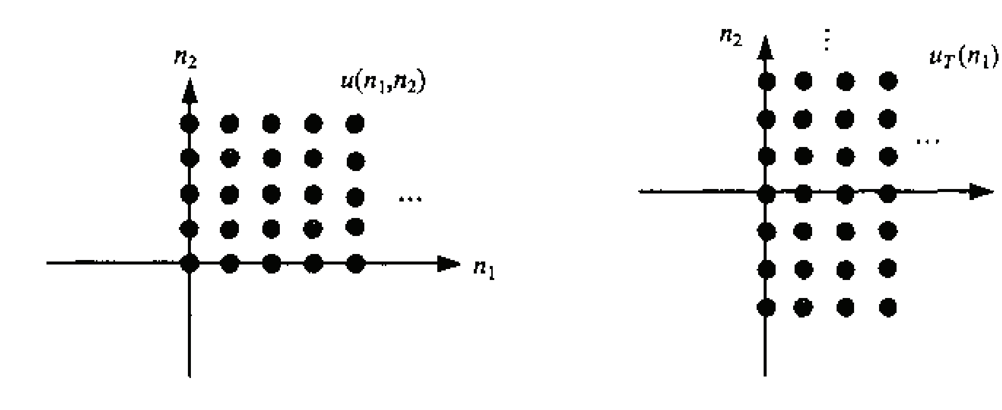


图 2.1-3 单位阶跃序列 $u(n_1, n_2)$ 图 2.1-4 单位阶跃序列 $u_T(n_1)$

- **指数序列**

指数序列的表示形式为 $x(n_1,n_2)=A\alpha^{n_1}\beta^{n_2}$，它对于系统分析相当重要。

- **可分离序列**

一个二维序列 $x(n_1,n_2)$ 如果可以表示为

$$x(n_1,n_2)=f(n_1)g(n_2) \tag{2.1-8}$$

则称其为可分离序列。此时 $f(n_1)$ 仅为 n_1 的函数，且 $g(n_2)$ 仅为 n_2 的函数。虽然可以将 $f(n_1)$ 及 $g(n_2)$ 视为二维序列，但将其视为一维序列会较为方便。因此，我们使用记号 $f(n_1)$ 及 $g(n_2)$ 而非 $f_T(n_1)$ 及 $g_T(n_2)$。

脉冲序列为一可分离序列，因为 $\delta(n_1,n_2)$ 可表示为

$$\delta(n_1,n_2)=\delta(n_1)\delta(n_2) \tag{2.1-9}$$

此时 $\delta(n_1)$ 及 $\delta(n_2)$ 为一维脉冲。单位阶跃序列 $u(n_1,n_2)$ 亦为一可分离序列，因 $u(n_1,n_2)$ 可表示为

$$u(n_1,n_2)=u(n_1)u(n_2) \tag{2.1-10}$$

此时 $u(n_1)$ 及 $u(n_2)$ 为一维单位阶跃序列。另一可分离序列的例子为 $a^{n_1}b^{n_2}+b^{n_1+n_2}$，可以写成 $(a^{n_1}+b^{n_1})b^{n_2}$。

- **周期序列**

如果 $x(n_1,n_2)$ 满足下述条件：对于所有 (n_1,n_2) 而言

$$x(n_1,n_2)=x(n_1+N_1,n_2)=x(n_1,n_2+N_2) \tag{2.1-11}$$

则称 $x(n_1,n_2)$ 为具有周期 $N_1\times N_2$ 的周期序列，此处 N_1 与 N_2 均为正整数。例如，$\cos\left[\pi n_1+\left(\frac{\pi}{2}\right)n_2\right]$ 为一具有周期 2×4 的周期序列，因为对于所有 (n_1,n_2)，

$$\cos\left[\pi n_1+\left(\frac{\pi}{2}\right)n_2\right]=\cos\left[\pi(n_1+2)+\left(\frac{\pi}{2}\right)n_2\right]=\cos\left[\pi n_1+\left(\frac{\pi}{2}\right)(n_2+4)\right] \quad (2.1\text{-}12)$$

序列 $\cos(n_1+n_2)$ 并非具有周期性，因为对于所有 (n_1,n_2) 而言，$\cos(n_1+n_2)$ 并不能表示为 $\cos[(n_1+N_1)+n_2]=\cos[n_1+(n_2+N_2)]$ 的形式，式中 N_1 与 N_2 为任意非零整数。

2.2　线性与线性时不变系统

若对于任何给定的输入而言都具有唯一的输出，则此一输入输出的关系称为系统。一个输入为 $x(n_1,n_2)$，输出为 $y(n_1,n_2)$ 的系统 T 表示为

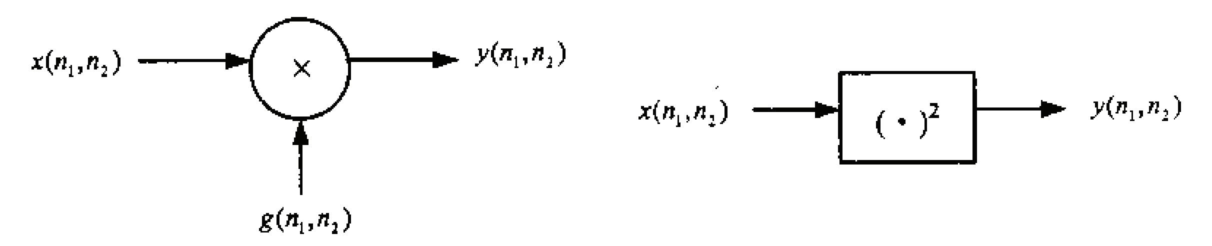
$$y(n_1,n_2)=T[x(n_1,n_2)] \quad (2.2\text{-}1)$$

在没有任何限制的状况下，描述一个系统需要一个完整的输入输出关系。若已知一组输入所对应的系统输出，通常并不能确知此一系统对另一组输入的输出为何。但若加入线性 (linearity) 与时不变性 (shift invariance) 的限制时，系统的描述可大大简化。值得庆幸的是，实际上许多系统都可以近似为线性与时不变性。

系统的线性定义为

$$\text{线性} \Leftrightarrow T[ax_1(n_1,n_2)+bx_2(n_1,n_2)]=ay_1(n_1,n_2)+by_2(n_1,n_2) \quad (2.2\text{-}2)$$

此处 $T[x_1(n_1,n_2)]=y_1(n_1,n_2)$，$T[x_2(n_1,n_2)]=y_2(n_1,n_2)$，$a$ 与 b 为任意纯量常数，$A\Leftrightarrow B$ 表示 A 包含 B 且 B 包含 A。上述条件也称为可叠加原理 (principle of superposition)，图 2.2-1 中的线性系统及图 2.2-2 中的非线性系统可以说明上述概念。图 2.2-1 中线性系统的线性以及图 2.2-2 中非线性系统的非线性均可用 (2.2-2) 式轻易地验证。



$y(n_1,n_2)=T[x(n_1,n_2)]=x(n_1,n_2)g(n_1,n_2)$

图 2.2-1　线性时不变系统实例

$y(n_1,n_2)=T[x(n_1,n_2)]=x^2(n_1,n_2)$

图 2.2-2　非线性时不变系统实例

系统的时不变性或称空不变性 (space invariance) 定义如下

$$\text{时不变性} \Leftrightarrow T[x(n_1-m_1,n_2-m_2)]=y(n_1-m_1,n_2-m_2) \quad (2.2\text{-}3)$$

此处 $y(n_1,n_2)=T[x(n_1,n_2)]$ 且 m_1 及 m_2 为整数。如图 2.2-1 中的系统并非时不变性，因为

$$T[x(n_1-m_1,n_2-m_2)]=x(n_1-m_1,n_2-m_2)g(n_1,n_2) \quad (2.2\text{-}4)$$

且

$$y(n_1-m_1,n_2-m_2)=x(n_1-m_1,n_2-m_2)g(n_1-m_1,n_2-m_2) \tag{2.2-5}$$

而如图 2.2-2 中的系统则具有时不变性，因为

$$T[x(n_1-m_1,n_2-m_2)]=x^2(n_1-m_1,n_2-m_2) \tag{2.2-6}$$

且

$$y(n_1-m_1,n_2-m_2)=x^2(n_1-m_1,n_2-m_2) \tag{2.2-7}$$

考虑一线性系统 T。使用 (2.1-2) 式及 (2.2-2) 式，我们可以将输入 $x(n_1,n_2)$ 的输出 $y(n_1,n_2)$ 表示为

$$\begin{aligned} y(n_1,n_2)=T[x(n_1,n_2)] &= T\left[\sum_{k_1=-\infty}^{\infty}\sum_{k_2=-\infty}^{\infty}x(k_1,k_2)\delta(n_1-k_1,n_2-k_2)\right] \\ &= \sum_{k_1=-\infty}^{\infty}\sum_{k_2=-\infty}^{\infty}x(k_1,k_2)T[\delta(n_1-k_1,n_2-k_2)] \end{aligned} \tag{2.2-8}$$

从上式可知：借助系统对于 $\delta(n_1,n_2)$ 及其时延 $\delta(n_1-k_1,n_2-k_2)$ 的响应可以完整地描述一个线性系统的特性。如果对于所有 k_1 与 k_2 的整数值，$T[\delta(n_1-k_1,n_2-k_2)]$ 为已知，则对应于任意输入 $x(n_1,n_2)$ 的线性系统输出均可由 (2.2-8) 式得到。但对非线性系统而言，对于所有 k_1 与 k_2 的整数值，若 $T[\delta(n_1-k_1,n_2-k_2)]$ 为已知，则当输入 $x(n_1,n_2)$ 为 $2\delta(n_1,n_2)$，$\delta(n_1,n_2)+\delta(n_1-1,n_2)$，或是其他许多序列时、我们无法得知其系统输出。

如果对一线性系统 T 加上时不变性的额外限制，则系统特性将可更进一步简化。假设我们将输入为 $\delta(n_1,n_2)$ 的系统 T 的响应以 $h(n_1,n_2)$ 表示为

$$h(n_1,n_2)=T[\delta(n_1,n_2)] \tag{2.2-9}$$

则从 (2.2-3) 式及 (2.2-9) 式得到

$$h(n_1-k_1,n_2-k_2)=T[\delta(n_1-k_1,n_2-k_2)] \tag{2.2-10}$$

其中 T 为一时不变系统。对于一线性时不变 (linear and shift-invariant, LSI) 系统而言，借助 (2.2-8) 式及 (2.2-10) 式可得其输入输出的关系为

$$y(n_1,n_2)=T[x(n_1,n_2)]=\sum_{k_1=-\infty}^{\infty}\sum_{k_2=-\infty}^{\infty}x(k_1,k_2)h(n_1-k_1,n_2-k_2) \tag{2.2-11}$$

上式由脉冲响应 $h(n_1,n_2)$ 完整地描述了 LSI 系统的特性。对于一个 LSI 系统而言，只知 $h(n_1,n_2)$ 便可确知任意输入时的输出为何。上式亦被称为卷积 (convolution)。

2.3 卷　积

卷积算子常以“*”表示，因此对于一个 LSI 系统，

$$
\begin{aligned}
y(n_1,n_2) &= x(n_1,n_2)*h(n_1,n_2) \\
&= \sum_{k_1=-\infty}^{\infty}\sum_{k_2=-\infty}^{\infty} x(k_1,k_2)h(n_1-k_1,n_2-k_2)
\end{aligned}
\tag{2.3-1}
$$

值得注意的是：在 LSI 系统中扮演重要角色的脉冲响应 $h(n_1,n_2)$，在非线性系统或非时变系统中失去它重要的意义。

(2.3-1) 式的卷积可用图形法求值。其步骤如下：

(1) 翻折：将 $h(k_1,\ k_2)$ 对 (0,0) 翻折以得到 $h(-k_1,\ -k_2)$。

(2) 移位：将 $h(-k_1,\ -k_2)$ 向右及向上分别移动 n_1 及 n_2 单位，以得到 $h(n_1-k_1,n_2-k_2)$。注意：若 n_1 为负值则向左移动，同理若 n_2 为负值则向下移动。

(3) 相乘：将 $x(k_1,k_2)$ 与 $h(n_1-k_1,n_2-k_2)$ 相乘得到乘积序列。

(4) 相加求和：将所有乘积序列相加以得到 $y(n_1,n_2)$。

图 2.3-1 以一例子说明上述步骤。

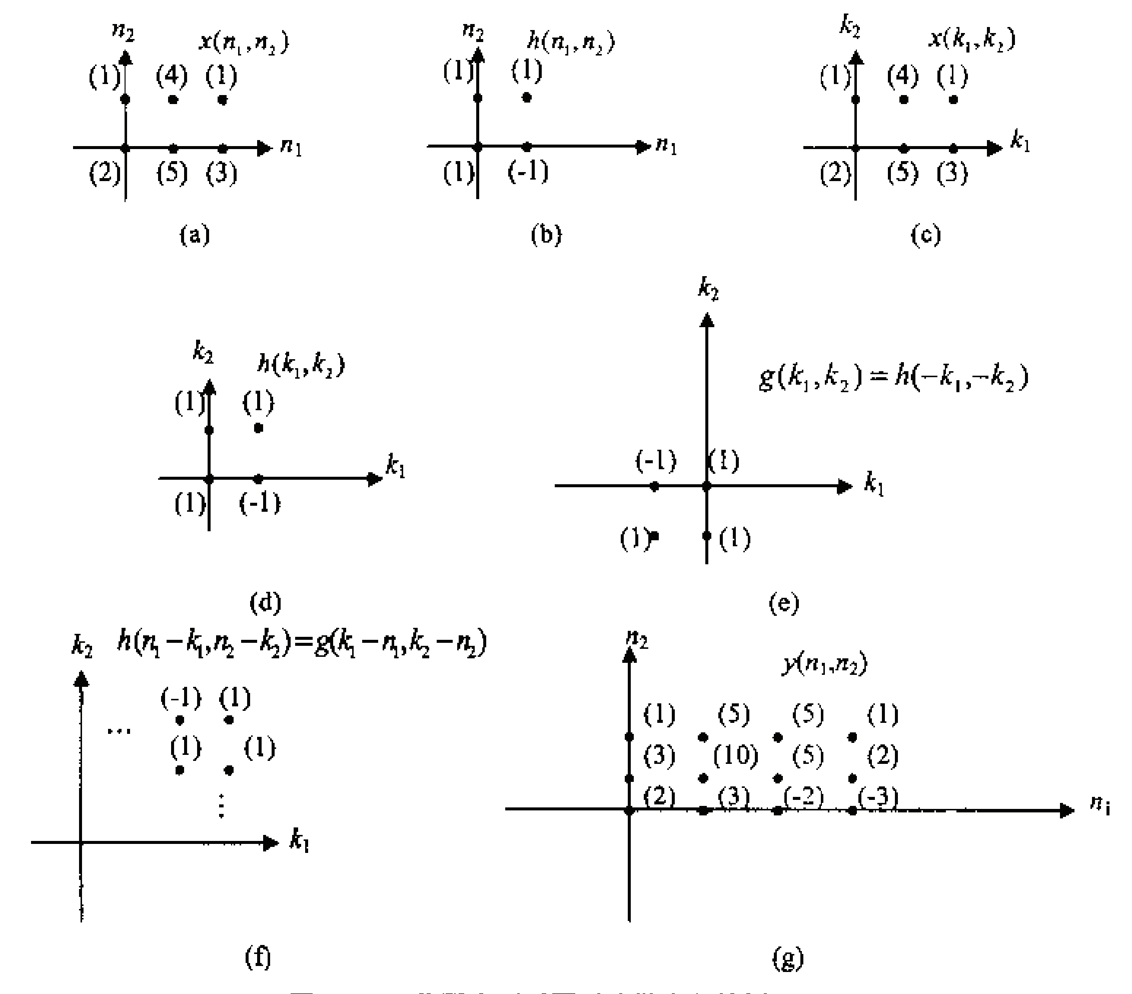


图 2.3-1　将两序列以图形法做卷积的例子

在 (2.3-1) 式中的卷积算子有一些一维结果的直接延伸。以下列举一些较重要的结果。

交换律 (commutativity)

$$x(n_1,n_2)*y(n_1,n_2)=y(n_1,n_2)*x(n_1,n_2) \tag{2.3-2}$$

结合律 (associativity)

$$[x(n_1,n_2)*y(n_1,n_2)]*z(n_1,n_2)=x(n_1,n_2)*[y(n_1,n_2)*z(n_1,n_2)] \tag{2.3-3}$$

分配律 (distributivity)

$$x(n_1,n_2)*[y(n_1,n_2)+z(n_1,n_2)]=[x(n_1,n_2)*y(n_1,n_2)]+[x(n_1,n_2)*z(n_1,n_2)] \tag{2.3-4}$$

与时延脉冲的卷积 (convolution with shifted impulse)

$$x(n_1,n_2)*\delta(n_1-m_1,n_2-m_2)=x(n_1-m_1,n_2-m_2) \tag{2.3-5}$$

交换律叙述了当输入与脉冲响应互换角色时，LSI 系统的输出并不受影响。结合律叙述了两个脉冲响应为 $h_1(n_1,n_2)$ 及 $h_2(n_1,n_2)$ 的 LSI 系统串接时，与一个脉冲响应为 $h_1(n_1,n_2)*h_2(n_1,n_2)$ 的 LSI 系统具有同样的输出输入关系。分配律叙述了两个脉冲响应为 $h_1(n_1,n_2)$ 及 $h_2(n_1,n_2)$ 的 LSI 系统平行结合时，与一个脉冲响应为 $h_1(n_1,n_2)+h_2(n_1,n_2)$ 的 LSI 系统具有同样的输出输入关系。

2.4 矩阵运算

2.4-1 向量与矩阵

一维或二维序列经常用向量或矩阵的形式表示。一个包含 N 个元素的列向量 $\boldsymbol{x}$ 表示为

$$\boldsymbol{x}=\{x(n)\}=\begin{bmatrix} x(1) \\ x(2) \\ \vdots \\ x(N) \end{bmatrix} \tag{2.4-1}$$

向量 $\boldsymbol{x}$ 的第 n 个元素表示成 $x(n)$，x_n，或 $[x]_n$。除非另外指明，否则所有的向量皆为列向量。一个大小为 N 的列向量称为一个 $N\times 1$ 向量，大小为 N 的行向量称为一个 $1\times N$ 向量。

一个大小为 $M\times N$ 的矩阵 $\boldsymbol{A}$ 具有 M 行与 N 列，定义为

$$\boldsymbol{A}=\{a(m,n)\}=\begin{bmatrix} a(1,1) & a(1,2) & \cdots & a(1,N) \\ a(2,1) & \cdots & \cdots & \cdots \\ \vdots & \vdots & \vdots & \vdots \\ a(M,1) & a(M,2) & \cdots & a(M,N) \end{bmatrix} \tag{2.4-2}$$

矩阵 $\boldsymbol{A}$ 中第 m 行及第 n 列的元素可写成 $[\boldsymbol{A}]_{m,n}=a(m,n)=a_{m,n}$。$\boldsymbol{A}$ 的第 n 列表示成 $\boldsymbol{a}_n$，其第 m 个元素写成 $a_n(m)=a(m,n)$。当矩阵的起始指标不是(1，1)时，我们以如下方式表示

$$\boldsymbol{A}=\{a(m,n)|\ 0\leqslant m,n\leqslant N-1\} \tag{2.4-3}$$

上式将一 $N\times N$ 矩阵以起始指标 (0，0) 重新表示。

在二维空间中，通常将一幅图像视为一个矩阵。这种矩阵表示法仅是将传统二维直角坐标系作一 90°的顺时针方向旋转

$$\{a(m,n)\}=\begin{bmatrix}2 & -1 & -3\\ 4 & 0 & 5\\ 1 & 2 & 3\end{bmatrix}\Rightarrow \quad \boldsymbol{A}=\begin{bmatrix}1 & 4 & 2\\ 2 & 0 & -1\\ 3 & 5 & -3\end{bmatrix}$$

2.4-2　矩阵重要的术语及性质

在此假设读者对线性代数有初步认识，本节只针对图像处理上较常用的部分做重点整理。

矩阵 $\boldsymbol{A}$ 的转置表为 $\boldsymbol{A}^{\mathrm{T}}$，其元素为

$$[\boldsymbol{A}^{\mathrm{T}}]_{m,n}=a_{n,m} \tag{2.4-4}$$

同理 $\boldsymbol{A}$ 的厄米特 (Hermitian) 转置 (或共轭转置) 表示为 $\boldsymbol{A}^{\mathrm{H}}$ 或 $\boldsymbol{A}^{*\mathrm{T}}$，其元素为

$$[\boldsymbol{A}^{\mathrm{H}}]_{m,n}=a_{n,m}^{*} \tag{2.4-5}$$

若 $\boldsymbol{A}^{\mathrm{H}}=\boldsymbol{A}$，则方形矩阵 $\boldsymbol{A}$ 称为厄米特的。

以下列举一些矩阵的共轭与转置规则

1. $\boldsymbol{A}^{*\mathrm{T}}=[\boldsymbol{A}^{\mathrm{T}}]^{*}$
2. $[\boldsymbol{AB}]^{\mathrm{T}}=\boldsymbol{B}^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}$
3. $[\boldsymbol{A}^{-1}]^{\mathrm{T}}=[\boldsymbol{A}^{\mathrm{T}}]^{-1}$
4. $[\boldsymbol{AB}]^{*}=\boldsymbol{A}^{*}\boldsymbol{B}^{*}$

- **矩阵的迹** (trace)

一个 $N\times N$ 方形矩阵 $\boldsymbol{A}$ 的迹为其主对角线上元素之和，表示成

$$\mathrm{tr}[\boldsymbol{A}]=\sum_{i=1}^{N}a(i,\ i) \tag{2.4-6}$$

若 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 均为方形矩阵，则

$$\mathrm{tr}[\boldsymbol{AB}]=\mathrm{tr}[\boldsymbol{BA}] \tag{2.4-7}$$

- **向量的范数** (norm)

一个 $N\times1$ 大小的向量 $\boldsymbol{x}$ 的欧几里德 (Euclidean) 向量范数为一定义如下的纯量

$$\|\boldsymbol{x}\|=\sqrt{\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}} \tag{2.4-8}$$

- **矩阵的范数**

一个 $N\times N$ 大小的矩阵 $\boldsymbol{A}$ 的欧几里德矩阵范数为一定义如下的纯量

$$\|\boldsymbol{A}\|=\sqrt{\mathrm{tr}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})} \tag{2.4-9}$$

- **矩阵的秩** (rank)

对一 $N\times N$ 的矩阵 $\boldsymbol{A}$，若其最大的非奇异 (nonsingular) 方形子矩阵为 $R\times R$，则矩阵 $\boldsymbol{A}$ 的秩为 R。它也等于线性无关的行或列的个数。若矩阵 $\boldsymbol{A}$ 及 $\boldsymbol{B}$ 为非奇异矩阵，而 $\boldsymbol{C}$ 为任意矩阵，则

$$\mathrm{rank}(\boldsymbol{C})=\mathrm{rank}(\boldsymbol{AC})=\mathrm{rank}(\boldsymbol{CA})=\mathrm{rank}(\boldsymbol{ACB}) \tag{2.4-10}$$

矩阵 $\boldsymbol{A}$ 及 $\boldsymbol{B}$ 之乘积的秩满足下列关系

$$\begin{aligned}&\mathrm{rank}(\boldsymbol{AB})\leqslant\mathrm{rank}(\boldsymbol{A})\\&\mathrm{rank}(\boldsymbol{AB})\leqslant\mathrm{rank}(\boldsymbol{B})\end{aligned} \tag{2.4-11}$$

矩阵 $\boldsymbol{A}$ 及 $\boldsymbol{B}$ 之和的秩则有如下关系

$$\mathrm{rank}(\boldsymbol{A}+\boldsymbol{B})\leqslant\mathrm{rank}(\boldsymbol{A}) \tag{2.4-12}$$

- **向量内积**

$N\times1$ 向量 $\boldsymbol{u}$ 及 $\boldsymbol{v}$ 的内积为纯量

$$\alpha=\boldsymbol{v}^{\mathrm{T}}\boldsymbol{u} \tag{2.4-13}$$

- **向量外积**

$M\times1$ 向量 $\boldsymbol{v}$ 与 $N\times1$ 向量 $\boldsymbol{u}$ 的外积为一矩阵

$$\boldsymbol{A}=\boldsymbol{v}\boldsymbol{u}^{\mathrm{T}} \tag{2.4-14}$$

- **二次型** (quadratic form)

一个 $N\times1$ 实数向量 $\boldsymbol{x}$ 的二次型定义为一纯量

$$\alpha=\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x} \tag{2.4-15}$$

其中 $\boldsymbol{A}$ 为 $N\times N$ 矩阵。通常 $\boldsymbol{A}$ 选定为一对称矩阵。

2.4-3　矩阵实例

● **Toeplitz 与循环矩阵**

Toeplitz矩阵 $\boldsymbol{T}$ 为一在其主要及次要对角线上皆为固定元素的矩阵。这代表元素 $t(m,n)$ 仅由 $m-n$ 这个差来决定，亦即 $t(m,n)=t_{m-n}$，如此一来一个 $N\times N$ 的 Toeplitz 矩阵便有如下的形式

$$\boldsymbol{T}=\begin{bmatrix} t_0 & t_{-1} & & \cdots & t_{-N+1} \\ t_1 & t_0 & t_{-1} & & t_{-N+2} \\ t_2 & & \ddots & \ddots & \\ \vdots & & & & t_{-1} \\ t_{N-1} & \cdots & t_2 & t_1 & t_0 \end{bmatrix} \tag{2.4-16}$$

且借助 $(2N-1)$ 个元素 $\{t_k, -N+1\leqslant k\leqslant N-1\}$ 来定义。

若矩阵 $\boldsymbol{C}$ 的行（或列）为前一行（或列）的循环偏移，也就是

$$\boldsymbol{C}=\begin{bmatrix} c_0 & c_1 & c_2 & \cdots & c_{N-1} \\ c_{N-1} & c_0 & c_1 & & c_{N-2} \\ \vdots & & \ddots & \ddots & \vdots \\ c_2 & & & & c_1 \\ c_1 & c_2 & \cdots & c_{N-1} & c_0 \end{bmatrix} \tag{2.4-17}$$

则称 $\boldsymbol{C}$ 为循环 (circulant) 矩阵，注意到 $\boldsymbol{C}$ 同时亦为 Toeplitz 矩阵，且

$$c(m,n)=c_{((m-n)\ \mathrm{modulo}\ N)} \tag{2.4-18}$$

● **正交及么正矩阵**

正交 (orthogonal) 矩阵定义为其逆矩阵等于其转置矩阵，也就是说若

$$\boldsymbol{A}^{-1}=\boldsymbol{A}^{\mathrm{T}} \tag{2.4-19}$$

或

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}=\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}=\boldsymbol{I} \tag{2.4-20}$$

则 $\boldsymbol{A}$ 为正交。

若矩阵的逆矩阵等于其共轭矩阵，则称其为么正 (unitary) 矩阵，也就是说

$$\boldsymbol{A}^{-1}=\boldsymbol{A}^{*\mathrm{T}} \tag{2.4-21}$$

或

$$\boldsymbol{A}\boldsymbol{A}^{*\mathrm{T}}=\boldsymbol{A}^{*\mathrm{T}}\boldsymbol{A}=\boldsymbol{I} \tag{2.4-22}$$

一实数正交矩阵同时也是么正矩阵，但一么正矩阵却不需为正交。由上述的定义可知一 $N\times N$ 么正矩阵的行（或列）为正交且在一 N 维向量空间形成一组标准正交基。

- **恒正性质与二次形式**

若一 $N\times N$ 的厄米特矩阵 $\boldsymbol{A}$ 的二次形式

$$Q=\boldsymbol{x}^{*\mathrm{T}}\boldsymbol{A}\boldsymbol{x},\quad \forall \boldsymbol{x}\neq \boldsymbol{0} \tag{2.4-23}$$

为正 (>0) 或非负 (≥ 0)，则将其分别称为正定 (positive definite) 及半正定 (positive semidefinite)。同理，当 Q 小于零或小于等于零时，$\boldsymbol{A}$ 称为负定或半负定。不满足上述条件的矩阵为不定的 (indefinite)。

若 $\boldsymbol{A}$ 为一对称正定 (非负) 矩阵，则其所有特征值 $\{\lambda_k\}$ 皆为正 (非负) 数且 $\boldsymbol{A}$ 的行列式值满足下列不等式

$$|\boldsymbol{A}|=\prod_{k=1}^{N}\lambda_k\leq\prod_{k=1}^{N}a(k,k) \tag{2.4-24}$$

- **对角形式**

对任意厄米特矩阵 $\boldsymbol{R}$ 都存在一么正矩阵 $\boldsymbol{\Phi}$，使得

$$\boldsymbol{\Phi}^{*\mathrm{T}}\boldsymbol{R}\boldsymbol{\Phi}=\boldsymbol{\Lambda} \tag{2.4-25}$$

此处 $\boldsymbol{\Lambda}$ 为一包含 $\boldsymbol{R}$ 的特征值的对角 (diagonal) 矩阵。上式的另一种形式为

$$\boldsymbol{R}\boldsymbol{\Phi}=\boldsymbol{\Phi}\boldsymbol{\Lambda} \tag{2.4-26}$$

它是下列特征值方程式的一个集合。

$$\boldsymbol{R}\boldsymbol{\Phi}_k=\lambda_k\boldsymbol{\Phi}_k\qquad k=1,\ 2,\ \cdots,\ N \tag{2.4-27}$$

其中 $\{\lambda_k\}$ 及 $\{\boldsymbol{\Phi}_k\}$ 分别为 $\boldsymbol{R}$ 的特征值及特征向量。

- **分块矩阵**

矩阵的元素本身亦为矩阵者称为分块矩阵 (block matrix)。例如

$$\boldsymbol{A}=\begin{bmatrix}\boldsymbol{A}_{1,1} & \boldsymbol{A}_{1,2} & \cdots & \boldsymbol{A}_{1,n}\\ \boldsymbol{A}_{2,1} & \boldsymbol{A}_{2,2} & \cdots & \boldsymbol{A}_{2,n}\\ \vdots & \vdots & \vdots & \vdots\\ \boldsymbol{A}_{m,1} & \boldsymbol{A}_{m,2} & \cdots & \boldsymbol{A}_{m,n}\end{bmatrix} \tag{2.4-28}$$

当分块结构为 Toeplitz (即 $\boldsymbol{A}_{m,n}=\boldsymbol{A}_{m-n}$) 或循环 (即 $\boldsymbol{A}_{m,n}=\boldsymbol{A}_{((m-n)\ \mathrm{modulo}\ N)}$) 时，则矩阵 $\boldsymbol{A}$ 分别称为分块 Toeplitz 或分块循环。此外，若每一分块本身也都是 Toeplitz (或循环的)，则 $\boldsymbol{A}$ 称为双重分块 Toeplitz (或双重分块循环)。

分块矩阵可用来简化许多图像处理中的分析工作。例如，二维的卷积就可用简单的分块

矩阵运算达成。考虑先前图 2.3-1 所示的例子，我们可得如下的结果：

$$\boldsymbol{y}=\begin{bmatrix}\boldsymbol{y}_0\\\boldsymbol{y}_1\\\boldsymbol{y}_2\end{bmatrix}=\begin{bmatrix}\boldsymbol{H}_0 & \boldsymbol{0}\\\boldsymbol{H}_1 & \boldsymbol{H}_0\\\boldsymbol{0} & \boldsymbol{H}_1\end{bmatrix}\begin{bmatrix}\boldsymbol{x}_0\\\boldsymbol{x}_1\end{bmatrix}=\boldsymbol{Hx} \tag{2.4-29}$$

其中 $\boldsymbol{y}_n$ 为输出 $y(n_1,n_2)$ 所形成的列向量，例如 $\boldsymbol{y}_0=\begin{bmatrix}2\\3\\-2\\-3\end{bmatrix}$； $\boldsymbol{x}_0=\begin{bmatrix}2\\5\\3\end{bmatrix}$， $\boldsymbol{x}_1=\begin{bmatrix}1\\4\\1\end{bmatrix}$，

$\boldsymbol{H}_0=\begin{bmatrix}1 & 0 & 0\\-1 & 1 & 0\\0 & -1 & 1\\0 & 0 & -1\end{bmatrix}$， $\boldsymbol{H}_1=\begin{bmatrix}1 & 0 & 0\\1 & 1 & 0\\0 & 1 & 1\\0 & 0 & 1\end{bmatrix}$。由 (2.4-29) 式可看出 $\boldsymbol{H}$ 为 3×2 分块 Toeplitz 矩阵。另外，当有周期序列做二维的卷积时可形成分块循环矩阵，请读者自行举例。

- **矩阵直接或 Kronecker 乘积**

一个 $M\times N$ 大小的矩阵 $\boldsymbol{A}$ 与 $P\times Q$ 大小的矩阵 $\boldsymbol{B}$ 所形成的右直接乘积 (direct product) 为一大小为 $PM\times QN$ 且定义如下的矩阵：

$$\boldsymbol{C}=\boldsymbol{A}\otimes\boldsymbol{B}=\{a(m,n)\boldsymbol{B}\}=\begin{bmatrix}a(1,1)\boldsymbol{B} & \cdots & a(1,N)\boldsymbol{B}\\a(2,1)\boldsymbol{B} & \cdots & a(2,N)\boldsymbol{B}\\\vdots & \vdots & \vdots\\a(M,1)\boldsymbol{B} & \cdots & a(M,N)\boldsymbol{B}\end{bmatrix} \tag{2.4-30}$$

同理，左直接乘积则可定义成 $\boldsymbol{C}=\boldsymbol{B}\otimes\boldsymbol{A}=\{b(p,q)\boldsymbol{A}\}$。本书将只考虑 (2.4-30) 式的右直接乘积 (以下简称直接乘积)。直接乘积主要用来从较小的矩阵依循环的方式产生较大的矩阵，例如第四章某些变换的基底所形成的矩阵就具有这种关系，其功能是表达一个较大图像分块的变换可循环分解成较小但运算较快的小分块变换。

一般而言 $\boldsymbol{A}\otimes\boldsymbol{B}\neq\boldsymbol{B}\otimes\boldsymbol{A}$。考虑如下的例子

$$\boldsymbol{A}=\begin{bmatrix}1 & 1\\-1 & 1\end{bmatrix}，\boldsymbol{B}=\begin{bmatrix}1 & 3\\2 & 4\end{bmatrix}$$

$$\boldsymbol{A}\otimes\boldsymbol{B}=\begin{bmatrix}1 & 3 & 1 & 3\\2 & 4 & 2 & 4\\-1 & -3 & 1 & 3\\-2 & -4 & 2 & 4\end{bmatrix}\neq\boldsymbol{B}\otimes\boldsymbol{A}=\begin{bmatrix}1 & 1 & 3 & 3\\-1 & 1 & -3 & 3\\2 & 2 & 4 & 4\\-2 & 2 & -4 & 4\end{bmatrix}$$

直接乘积有下列性质：

1. $(\boldsymbol{A}+\boldsymbol{B})\otimes\boldsymbol{C}=\boldsymbol{A}\otimes\boldsymbol{C}+\boldsymbol{B}\otimes\boldsymbol{C}$；

2. $(\boldsymbol{A}\otimes\boldsymbol{B})\otimes\boldsymbol{C}=\boldsymbol{A}\otimes(\boldsymbol{B}\otimes\boldsymbol{C})$；
3. $\alpha(\boldsymbol{A}\otimes\boldsymbol{B})=(\alpha\boldsymbol{A})\otimes\boldsymbol{B}=\boldsymbol{A}\otimes(\alpha\boldsymbol{B})$，其中 α 为纯量；
4. $(\boldsymbol{A}\otimes\boldsymbol{B})^{\mathrm{T}}=\boldsymbol{A}^{\mathrm{T}}\otimes\boldsymbol{B}^{\mathrm{T}}$；
5. $(\boldsymbol{A}\otimes\boldsymbol{B})^{-1}=\boldsymbol{A}^{-1}\otimes\boldsymbol{B}^{-1}$；
6. $(\boldsymbol{A}\otimes\boldsymbol{B})(\boldsymbol{C}\otimes\boldsymbol{D})=(\boldsymbol{AC})\otimes(\boldsymbol{BD})$；
7. $\mathrm{tr}(\boldsymbol{A}\otimes\boldsymbol{B})=\mathrm{tr}(\boldsymbol{A})\mathrm{tr}(\boldsymbol{B})$；
8. $\mathrm{rank}(\boldsymbol{A}\otimes\boldsymbol{B})=\mathrm{rank}(\boldsymbol{A})\mathrm{rank}(\boldsymbol{B})$；
9. 若 $\boldsymbol{A}$ 与 $\boldsymbol{B}$ 为么正，则 $\boldsymbol{A}\otimes\boldsymbol{B}$ 亦是。

性质 6 的结果对于发展含矩阵相乘的快速算法很有用，因为等式左边的运算量为 $O(N^6)+O(N^4)$ 而右边则只需 $O(N^4)$（设四个矩阵的大小均为 $N\times N$）。

2.5 纯量值对向量参数的最优化

在图像处理与许多其他的工程问题上常出现对一函数求最优化的问题，本节考虑此问题的数学背景。设 f 为实数 N 维向量参数 $\boldsymbol{x}$ 的纯量值函数，即 $f(\boldsymbol{x})$。梯度 $\nabla_{\boldsymbol{x}}f$ 定义为下列偏导数的向量

$$\nabla_{\boldsymbol{x}}f=\begin{bmatrix}\dfrac{\partial f}{\partial x_1}\\ \dfrac{\partial f}{\partial x_2}\\ \vdots\\ \dfrac{\partial f}{\partial x_N}\end{bmatrix} \tag{2.5-1}$$

其中 $x_1,x_2,\cdots,x_N$ 为 $\boldsymbol{x}$ 的分量。由 $\nabla_{\boldsymbol{x}}f=\boldsymbol{0}$ 可得 f 最大或最小所要的条件，进而得到最佳化的解。

若 f 为向量内积 $f=\boldsymbol{v}^{\mathrm{T}}\boldsymbol{x}$，其中 $\boldsymbol{v}$ 亦为实数向量，则很容易可证明

$$\nabla_{\boldsymbol{x}}(\boldsymbol{v}^{\mathrm{T}}\boldsymbol{x})=\nabla_{\boldsymbol{x}}(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{v})=\boldsymbol{v} \tag{2.5-2}$$

同理，若 f 为二次型 $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}$，则其梯度为

$$\nabla_{\boldsymbol{x}}(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x})=(\boldsymbol{A}+\boldsymbol{A}^{\mathrm{T}})\boldsymbol{x} \tag{2.5-3}$$

若再加上 $\boldsymbol{A}$ 为对称，则

$$\nabla_{\boldsymbol{x}}(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x})=2\boldsymbol{A}\boldsymbol{x} \tag{2.5-4}$$

- **有条件限制的最优化**

假设我们要将某个函数 $f(x_1,x_2)$ 最优化，但是有如下的条件限制

$$c(x_1,x_2)=0 \tag{2.5-5}$$

直接做法是解上式得到两个变量 x_1 及 x_2 的关系，例如

$$x_2 = g(x_1) \tag{2.5-6}$$

再将上式代人 f 中并将 $f(x_1, g(x_1))$ 最优化，现在这个问题已变成仅含单一变量且无条件的最优化问题。这个方法对某些简单问题可能有效，但一般而言并不太有用。较佳的解法是采用拉格朗日乘数法 (Lagrange multipliers)。

首先引人一个拉格朗日变量 μ 并形成称为拉格朗日的新函数

$$L(x_1, x_2, \mu) = f(x_1, x_2) + \mu c(x_1, x_2) \tag{2.5-7}$$

这样做是要将一个含两变量且有条件限制的最优化 (constrained optimization) 问题转化成含三个变量但无条件限制的最优化问题。使 $L(x_1, x_2, \mu)$ 最优化的必要条件为

$$\frac{\partial L}{\partial x_1} = \frac{\partial f}{\partial x_1} + \mu \frac{\partial c}{\partial x_1} = 0 \tag{2.5-8}$$

$$\frac{\partial L}{\partial x_2} = \frac{\partial f}{\partial x_2} + \mu \frac{\partial c}{\partial x_2} = 0 \tag{2.5-9}$$

$$\frac{\partial L}{\partial \mu} = c(x_1, x_2) = 0 \tag{2.5-10}$$

(2.5-10) 正是原始的限制。因为这组方程式使 L 最优化并同时强迫 c 为零，因而 f 为最佳，同时亦满足条件限制。

现在将此问题推广成实数向量变量 $\boldsymbol{x}$ 并假设有多个限制。例如有两个条件 $c_1(\boldsymbol{x}) = 0$ 及 $c_2(\boldsymbol{x}) = 0$，则其拉格朗日的函数为

$$L(\boldsymbol{x}, \mu_1, \mu_2) = f(\boldsymbol{x}) + \mu_1 c_1(\boldsymbol{x}) + \mu_2 c_2(\boldsymbol{x}) \tag{2.5-11}$$

而所要解的方程式变成

$$\nabla_{\boldsymbol{x}} L = \nabla_{\boldsymbol{x}} f(\boldsymbol{x}) + \mu_1 \nabla_{\boldsymbol{x}} c_1(\boldsymbol{x}) + \mu_2 \nabla_{\boldsymbol{x}} c_2(\boldsymbol{x}) \tag{2.5-12}$$

$$\frac{\partial L}{\partial \mu_1} = c_1(\boldsymbol{x}) = 0 \tag{2.5-13}$$

$$\frac{\partial L}{\partial \mu_2} = c_2(\boldsymbol{x}) = 0 \tag{2.5-14}$$

对更多的限制则可依此类推。

2.6　随机信号

本节将讨论描述图像信号的统计特性所常采用的度量方式。假设读者有随机变量与随机过程的基本概念，此处只对图像处理较相关的部分做简要整理。

一个大小 $N\times 1$ 的随机向量 $\boldsymbol{x}$ 的平均 (mean) 定义为

$$\boldsymbol{m}_{\boldsymbol{x}}=E\{\boldsymbol{x}\}=\begin{bmatrix} m_1 \\ m_2 \\ \vdots \\ m_N \end{bmatrix} \tag{2.6-1}$$

其中 m_i 为随机变量 x_i 的平均。其相关 (correlation) 矩阵定义为

$$\boldsymbol{R}_{\boldsymbol{x}\boldsymbol{x}}=E\left\{\boldsymbol{x}\boldsymbol{x}^{*\mathrm{T}}\right\} \tag{2.6-2}$$

其中第 k 行第 l 列的元素为 $E\left\{x_k x_l^*\right\}$。其协方差 (covariance) 矩阵定义为

$$\boldsymbol{C}_{\boldsymbol{x}\boldsymbol{x}}=E\left\{(\boldsymbol{x}-\boldsymbol{m}_{\boldsymbol{x}})(\boldsymbol{x}-\boldsymbol{m}_{\boldsymbol{x}})^{*\mathrm{T}}\right\} \tag{2.6-3}$$

其中此矩阵的元素为

$$c_{kl}=E\left\{(x_k-m_k)(x_l-m_l)^*\right\} \tag{2.6-4}$$

且

$$\begin{aligned} c_{kk} &= E\left\{\left|x_k-m_k\right|^2\right\}=\mathrm{Var}[x_k] \\ &= x_k \text{ 的方差 (variance)} \end{aligned} \tag{2.6-5}$$

由 (2.6-2) 及 (2.6-3) 两式可看出，相关矩阵与协方差矩阵均为厄米特矩阵，亦即

$$\boldsymbol{R}_{\boldsymbol{x}\boldsymbol{x}}=\boldsymbol{R}_{\boldsymbol{x}\boldsymbol{x}}^{*\mathrm{T}}, \quad \boldsymbol{C}_{\boldsymbol{x}\boldsymbol{x}}=\boldsymbol{C}_{\boldsymbol{x}\boldsymbol{x}}^{*\mathrm{T}} \tag{2.6-6}$$

此外这两个矩阵均为半正定；换言之，对任一复数向量 $\boldsymbol{a}$，它们有如下关系

$$\boldsymbol{a}^{*\mathrm{T}}\boldsymbol{R}_{\boldsymbol{x}\boldsymbol{x}}\boldsymbol{a}\geqslant 0, \quad \boldsymbol{a}^{*\mathrm{T}}\boldsymbol{C}_{\boldsymbol{x}\boldsymbol{x}}\boldsymbol{a}\geqslant 0 \tag{2.6-7}$$

且

$$\boldsymbol{R}_{\boldsymbol{x}\boldsymbol{x}}=\boldsymbol{C}_{\boldsymbol{x}\boldsymbol{x}}+\boldsymbol{m}_{\boldsymbol{x}}\boldsymbol{m}_{\boldsymbol{x}}^{*\mathrm{T}} \tag{2.6-8}$$

若 $\boldsymbol{R}_{\boldsymbol{xy}}=\boldsymbol{R}_{\boldsymbol{yx}}=\boldsymbol{0}$，则称 $\boldsymbol{x}$ 与 $\boldsymbol{y}$ 为正交，此时 $\boldsymbol{R}_{\boldsymbol{x}+\boldsymbol{y}}=\boldsymbol{R}_{\boldsymbol{x}\boldsymbol{x}}+\boldsymbol{R}_{\boldsymbol{y}\boldsymbol{y}}$；而若 $\boldsymbol{C}_{\boldsymbol{x}\boldsymbol{y}}=\boldsymbol{C}_{\boldsymbol{y}\boldsymbol{x}}=\boldsymbol{0}$，则想 $\boldsymbol{x}$ 与 $\boldsymbol{y}$ 为非相关 (uncorrelated)，此时 $\boldsymbol{C}_{\boldsymbol{x}+\boldsymbol{y}}=\boldsymbol{C}_{\boldsymbol{x}\boldsymbol{x}}+\boldsymbol{C}_{\boldsymbol{y}\boldsymbol{y}}$。

- **多变量高斯密度函数**

一个实数高斯随机 N 维向量 $\boldsymbol{x}$ 是由多变量 (multivariate) 正态 (normal) 或高斯密度函数所描述的向量。此密度函数为

$$f_{\boldsymbol{x}}(\boldsymbol{x})=\frac{1}{(2\pi)^{\frac{N}{2}}\left|\boldsymbol{C}_{\boldsymbol{x}\boldsymbol{x}}\right|^{\frac{1}{2}}}\mathrm{e}^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{m}_x)^{\mathrm{T}}\boldsymbol{C}_{xx}^{-1}(\boldsymbol{x}-\boldsymbol{m}_x)} \tag{2.6-9}$$

此函数由平均向量 $\boldsymbol{m}_{\boldsymbol{x}}$ 及其协方差矩阵 $\boldsymbol{C}_{\boldsymbol{x}\boldsymbol{x}}$ 可完整描述。

● **平稳过程**

对一个随机序列 $x(n)$，若任何部分序列 $\{x(l), 1 \le l \le k\}$ 与其平移后的序列 $\{x(l+m), 1 \le l \le k\}$ 对任何整数 m 与 k 而言都有相同的联合密度函数，则此序列为狭义平稳 (strict-sense stationary)。若

$$E\{x(n)\} = m_x(n) = \text{常数} \tag{2.6-10}$$

$$E\{x(k)x^*(l)\} = r_x(k,l) = r_x(k-l) \tag{2.6-11}$$

则 $x(n)$ 称为广义 (wide-sense) 平稳。(2.6-11)式显示 $\{x(n)\}$ 的协方差矩阵为 Toeplitz 矩阵。

● **马尔可夫过程**

对一随机过程 $x(n)$，若条件概率分布

$$Pr[x(n)|x(n-1), x(n-2), \cdots] = Pr[x(n)|x(n-1), \cdots, x(n-p)] \quad \forall n \tag{2.6-12}$$

则 $x(n)$ 称为马尔可夫-p (Markov-p) 或 p 阶马尔可夫过程。换言之，若"目前" $\{x(j), n-p \le j \le n-1\}$ 已知，则"过去" $\{x(j), j < n-p\}$ 以及"未来" $\{x(j), j \ge n\}$ 为独立的。一个马尔可夫-1 序列是最简单也最常被使用的过程，以下我们只考虑此状况。

一阶平稳马尔可夫序列 $x(n)$ 的协方差函数为 $r(n) = \rho^{|n|}$，$|\rho| < 1$。此函数常作为单色图像一列像素的协方差模型。对一个 $N \times 1$ 向量 $\boldsymbol{x} = \{x(n), 1 \le n \le N\}$，其协方差矩阵 $\boldsymbol{R}$ 为 $\{r(m,n) = r(m-n)\}$，亦即

$$\boldsymbol{R} = \begin{bmatrix} 1 & \rho & \cdots & \rho^{N-1} \\ \rho & 1 & \ddots & \rho^{N-2} \\ \rho^2 & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \rho \\ \rho^{N-1} & \rho^{N-2} & \cdots & 1 \end{bmatrix} \tag{2.6-13}$$

此为 Toeplitz 矩阵。事实上任何平稳序列的协方差数或自相关矩阵均为 Toeplitz 矩阵。

● **马尔可夫链**

当 $x(n)$ 只有可数的 (离散) 值，则马尔可夫过程称为马尔可夫链 (chain)。对任何数字信号处理 (包括数字图像) 都适用马尔可夫链，因为其随机序列都是以有限个离散值来表示的。

若 $x(n)$ 为满足下列条件的随机过程，则 $x(n)$ 为马尔可夫链

$$\begin{aligned} &Pr[x(n) = x_n | x(n-1) = x_{n-1}, x(n-2) = x_{n-2}, \cdots] \\ &= Pr[x(n) = x_n | x(n-1) = x_{n-1}] \end{aligned} \tag{2.6-14}$$

换言之，现在发生的事件只与最近发生的事件有关。以下是有关马尔可夫链的重点整理。假设 $x(n)$ 或 x_n 的可能值有 Q 个分别表成 $S_1, S_2, \cdots, S_Q$。当 $x(n) = S_i$ 时，我们说马尔可夫链在状态 i，而条件概率

$$P_{j|i}(n) = Pr[x(n) = S_j | x(n-1) = S_i] \tag{2.6-15}$$

则称为转移 (transition) 概率。设此转移概率为常数，亦即与 n 无关或 $P_{j|i}(n) = P_{j|i}$，则称此概率为齐次 (homogeneous) 转移概率。由此概率所构成的概率矩阵

$$\boldsymbol{P} = \begin{bmatrix} P_{1|1} & P_{2|1} & P_{3|1} & \cdots \\ P_{1|2} & P_{2|2} & P_{2|3} & \cdots \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} \tag{2.6-16}$$

或是其转置矩阵 $\boldsymbol{\Pi} = \boldsymbol{P}^{\mathrm{T}}$ 再加上初始状态概率可完整描述整个马尔可夫链，其中矩阵 $\boldsymbol{\Pi}$ 称为转移 (概率) 矩阵。

k 阶或 k 步转移概率定义为

$$P_{j|i}^{(k)} = Pr[x(n) = S_j | x(n-k) = S_i] \tag{2.6-17}$$

可证明 $\boldsymbol{P}^k$ 或 $\boldsymbol{\Pi}^k$ 含有所有 k 阶概率如下。若定义

$$\boldsymbol{p}(n) = \begin{bmatrix} p_1(n) \\ p_2(n) \\ \vdots \\ p_Q(n) \end{bmatrix} = \begin{bmatrix} Pr[x(n) = S_1] \\ Pr[x(n) = S_2] \\ \vdots \\ Pr[x(n) = S_Q] \end{bmatrix} \tag{2.6-18}$$

则显然

$$\boldsymbol{p}[n] = \boldsymbol{\Pi}\ \boldsymbol{p}[n-1] \tag{2.6-19}$$

迭代 k 次后可得

$$\boldsymbol{p}[n] = \boldsymbol{\Pi}^k \boldsymbol{p}[n-k] \tag{2.6-20}$$

此外

$$\boldsymbol{\Pi}^k = \boldsymbol{\Pi}^l \boldsymbol{\Pi}^{k-l}, \quad 0 < l < k \tag{2.6-21}$$

故

$$P_{j|i}^{(k)} = \sum_{q=1}^{Q} P_{j|q}^{(l)} P_{q|i}^{(k-l)}, \quad 0 < l < k \tag{2.6-22}$$

此即 Chapman-Kolmogorov 方程式。

当某个马尔可夫链的转移概率满足某些条件时，此马尔可夫链在大量的观察 (observation) 之后会有一个极限性质。考虑如图 2.6-1 所示的一个简单双状态马尔可夫链，几个 k 值的 k 阶转移概率的结果显示于表 2.6-1 中。由表中看出，不论起始状态为何，高阶转移概率都会收敛到一个定值。由(2.6-22)式可推得，此过程会达到某一个状态的概率将趋近于一个定值。因此该过程从起始状态起仅有“有限的记忆 (limited memory)”。

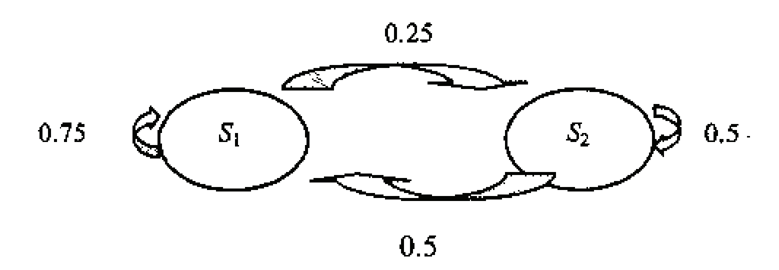


图 2.6-1　可达成极限概率的简单双状态马尔可夫链

表 2.6-1　双状态马尔可夫链的转移概率

k	1	2	3	4	5	6	7	8
$P_{1\|1}^{(k)}$	0.750	0.688	0.672	0.668	0.667	0.667	0.667	0.667
$P_{2\|1}^{(k)}$	0.250	0.312	0.328	0.332	0.333	0.333	0.333	0.333
$P_{1\|2}^{(k)}$	0.500	0.625	0.656	0.664	0.666	0.667	0.667	0.667
$P_{2\|2}^{(k)}$	0.500	0.375	0.344	0.336	0.334	0.333	0.333	0.333

● **离散随机场**

当二维序列的每个样本点都是随机变量时，我们称之为离散随机场 (random field)。先前关于一维随机信号的探讨都可推广到二维随机场。

若一个随机场的协方差函数可写成两个一维序列的协方差函数的乘积，则此协方差函数称为可分离的。常用于图像处理中的一个可分离平稳协方差函数为

$$r(k,l;k',l')=r(k,k')r(l,l')=\sigma^2\rho_1^{|k-k'|}\rho_2^{|l-l'|},\qquad |\rho_1|<1,|\rho_2|<1 \tag{2.6-23}$$

其中σ^2代表随机场的方差数，而ρ_1与ρ_2分别代表k方向与l方向坐标差一个像素间的相关性。对许多图像而言，$\rho_1=\rho_2=0.95$是一个很好的估测。

另一个被认为对许多图像而言更实际的协方差函数是

$$r(k,l;k',l')=\sigma^2\exp\left\{-\sqrt{\alpha_1(k-k')^2+\alpha_2(l-l')^2}\right\} \tag{2.6-24}$$

当$\alpha_1=\alpha_2=\alpha^2$时，$r(k,l;k',l')$变成欧几里德距离的函数，亦即

$$r(k,l;k',l')=\sigma^2\mathrm{e}^{-\alpha d} \tag{2.6-25}$$

其中$d=\sqrt{(k-k')^2+(l-l')^2}$。此函数具有各向同性 (isotropic) 或环形对称 (circularly symmetric)的特性。

习　题

1. 绘出下列序列

(1) $\delta(n_1+2,n_2-3)+2\delta(n_1,-n_2+2)$

(2) $\delta_T(n_1)u(n_1,n_2)$

(3) $\left(\frac{1}{2}\right)^{n_2}\delta_T(n_1+n_2)u(-n_1+1,-n_2)$

2. 试决定下列系统是否为 (1) 线性； (2) 时不变。

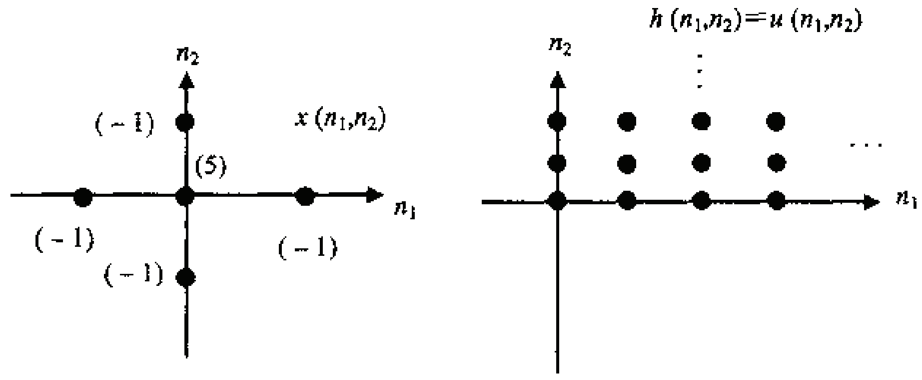
(1) $y(n_1,n_2)=T[x(n_1,n_2)]=\mathrm{e}^{3x(n_1,n_2)}$

(2) $y(n_1,n_2)=T[x(n_1,n_2)]=\sum_{k_1=n_1-3}^{n_1+6}x(k_1,n_2)$

(3) $y(n_1,n_2)=T[x(n_1,n_2)]=\sum_{k_2=0}^{n_2}x(n_1,k_2)$

3. 试计算下列二题的 $x(n_1,n_2)*h(n_1,n_2)$。

(1)



(2) $x(n_1,n_2)=\left(\frac{1}{2}\right)^{n_1}\left(\frac{1}{3}\right)^{n_2}u(n_1,n_2)$

$h(n_1,n_2)=u(n_1-1,n_2-2)$

4. 验证下列各点叙述：

(1) 一个循环矩阵为一 Toeplitz 矩阵，反之则不成立。

(2) 两个循环矩阵的乘积仍为一循环矩阵。

(3) 两个 Toeplitz 矩阵的乘积不一定为 Toeplitz 矩阵。

5. 试以图解法做以下序列的卷积：

(1) $x(n_1,n_2)=\delta(n_1,n_2)+2\delta(n_1-1,n_2)+2\delta(n_1,n_2-1)+3\delta(n_1-1,n_2-1)$

$h(n_1,n_2)=\delta(n_1,n_2)-\delta(n_1-1,n_2)+\delta(n_1,n_2-1)+2\delta(n_1-1,n_2-1)$

(2) $x(n_1,n_2)=\delta(n_1-1,n_2-1)+2\delta(n_1-2,n_2-1)+2\delta(n_1-1,n_2-2)+3\delta(n_1-2,n_2-2)$

$h(n_1,n_2)=\delta(n_1,n_2)+\delta(n_1-1,n_2)-\delta(n_1,n_2-1)+2\delta(n_1-1,n_2-1)$

6. 试以电脑程序验证上题的结果。

7. 设 $\boldsymbol{A}$ 为实数矩阵。若其固有值均为正值，则 $\boldsymbol{A}$ 为正定的。证明，当且仅当对每个非零的向量 $\boldsymbol{x}$，均有 $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}>0$，则 $\boldsymbol{A}$ 为正定的，其中

$$\boldsymbol{x}=\begin{bmatrix}x_1\\x_2\\\vdots\\x_N\end{bmatrix}$$

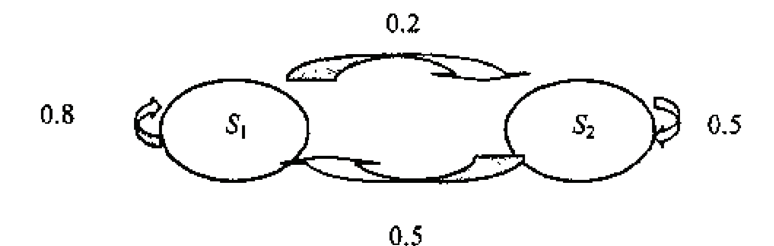
8. 试决定以下矩阵是否为么正或厄米特矩阵？

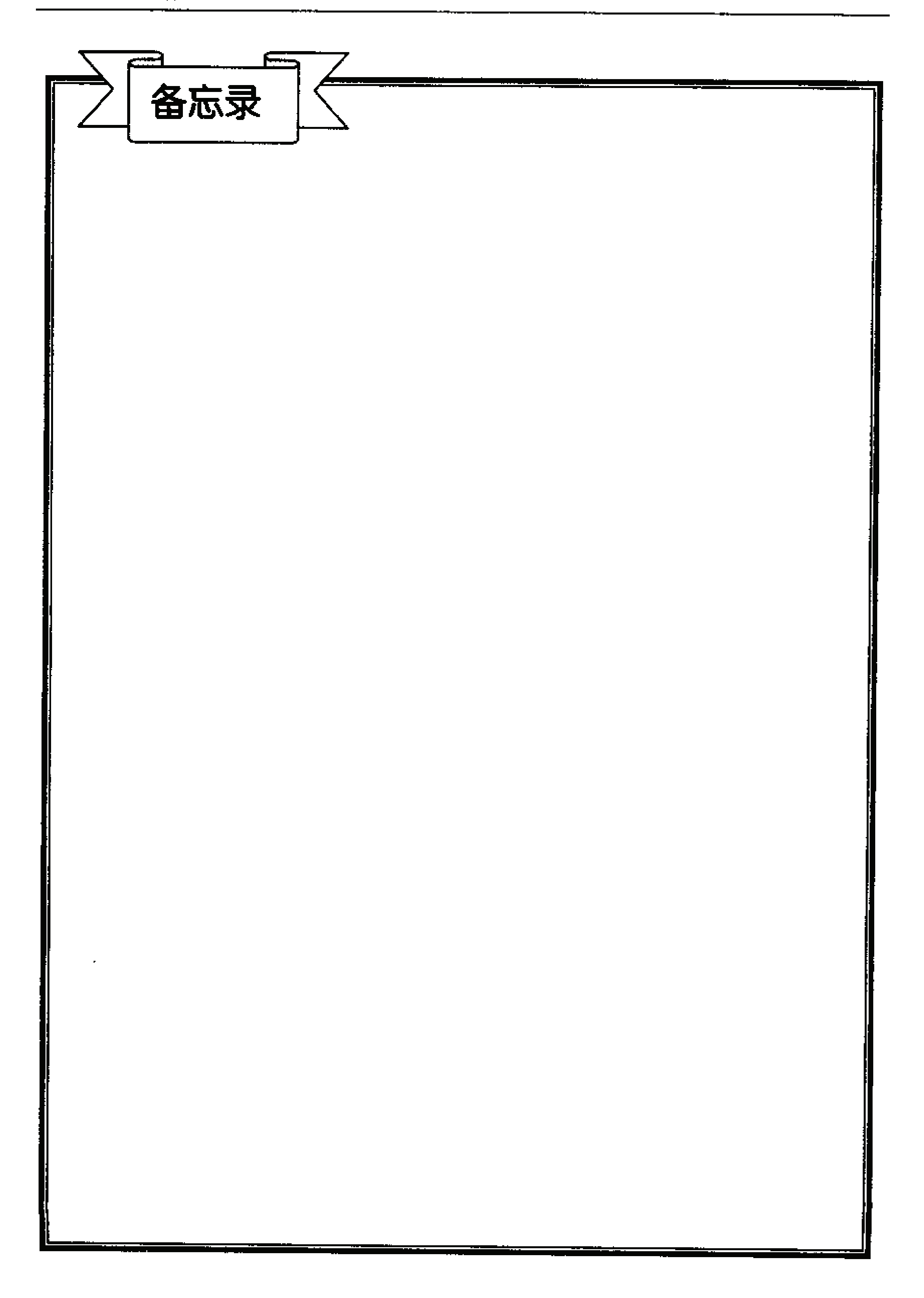
$$(1)\begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{i}{\sqrt{2}} & 0 \\ \frac{-1}{\sqrt{2}} & \frac{i}{\sqrt{2}} & 0 \\ 0 & 0 & i \end{bmatrix} \qquad (2)\begin{bmatrix} 3 & 2 & 0 \\ 2 & 0 & i \\ 0 & -i & 0 \end{bmatrix}$$

9. 若有某个非奇异矩阵 $\boldsymbol{P}$ 使得 $\boldsymbol{P}^{-1}\boldsymbol{APB}=\boldsymbol{B}$，则 $\boldsymbol{A}$ 相似 (similar) 于 $\boldsymbol{B}$。证明 $\boldsymbol{A}$ 与 $\boldsymbol{B}$ 有相同的秩。
10. 设 $\boldsymbol{A}$ 与 $\boldsymbol{B}$ 相似，证明其迹会相等。
11. 设 $\boldsymbol{A}$ 为可对角化的 $N\times N$ 矩阵。证明：其中有大小为 $N\times N$ 的矩阵 $\boldsymbol{P}_1,\boldsymbol{P}_2,\cdots,\boldsymbol{P}_N$ 使得
    (1) $\boldsymbol{A}=\lambda_1\boldsymbol{P}_1+\lambda_2\boldsymbol{P}_2+\cdots+\lambda_n\boldsymbol{P}_n$，其中 $\lambda_1,\lambda_2,\cdots,\lambda_n$ 为 $\boldsymbol{A}$ 的固有值。
    (2) $\boldsymbol{P}_j^2=\boldsymbol{P}_j$，其中 $j=1,2,\cdots,N$。
    (3) 若 $k\neq j$，则 $\boldsymbol{P}_k\boldsymbol{P}_j=\boldsymbol{0}$（$N\times N$ 的零矩阵）。
    (4) $\boldsymbol{P}_1+\boldsymbol{P}_2+\cdots+\boldsymbol{P}_N=\boldsymbol{I}_N$。
    [$\lambda_1\boldsymbol{P}_1+\lambda_2\boldsymbol{P}_2+\cdots+\lambda_N\boldsymbol{P}_N$ 称为 $\boldsymbol{A}$ 的分谱分解 (spectral decomposition)，而矩阵 $\boldsymbol{P}_j$ 称为投影 (projection)。]
    提示：先确定对某个 $\boldsymbol{P}$ 有 $\boldsymbol{P}^{-1}\boldsymbol{AP}=\boldsymbol{D}$，为一对角矩阵。将 $\boldsymbol{D}$ 写成 $\boldsymbol{D}=\lambda_1\boldsymbol{T}_{11}+\lambda_2\boldsymbol{T}_{22}+\cdots+\lambda_N\boldsymbol{T}_{NN}$，其中 $\boldsymbol{T}_{jj}$ 为在 jj 位置上为 1、其他位置上均为零的矩阵。
12. 举一简单例子显示：当有周期序列做二维的卷积时可形成分块循环矩阵。
13. 取两张 16×16 大小的图像数据或两个同等大小的任意矩阵，以电脑程序计算其直接乘积，并验证课文内容所列有关直接乘积的性质。
14. 考虑下列函数的极值的问题

$$f(\boldsymbol{x})=x_1^2+x_2^2$$

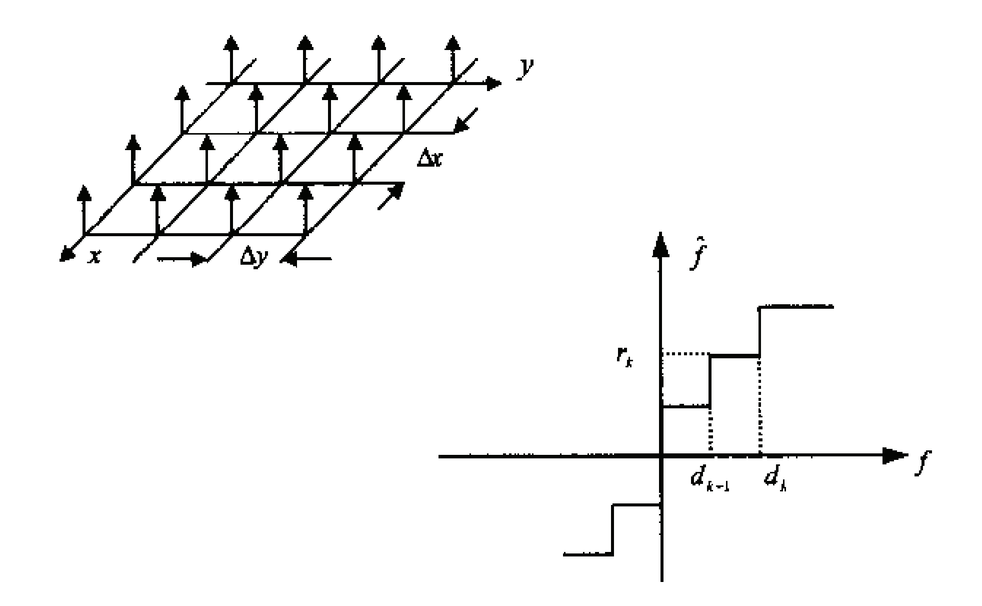
    其限制为 $x_1^2+2x_2^2-1=0$。求极大、极小值。
15. 证明一不相关随机变量序列的协方差矩阵为对角矩阵。
16. 证明实数平稳周期随机序列之一周期所形成的协方差矩阵为循环的。
17. 对下列双状态马尔可夫链做出类似表 2.6-1 的结果。




备忘录

第三章

取样与量化



在前二章曾提及，一幅灰度图像 (单色图像) 可看成是一个二维的连续函数 $f(x,y)$，其亮度为位置坐标 (x,y) 的连续函数。而一个数字图像是图像 $f(x,y)$ 在空间坐标和亮度都数字化的图像。从二维连续函数变成可用矩阵表示的数字图像，牵涉到在不同空间位置取出函数 (灰度) 值作为样本，并用一组整数值来表示这些样本的两个过程。前者称为取样，后者则称为量化，两者统称为数字化。图 3.0-1 为此数字化过程，其中 f_s 代表取样后的结果，m 与 n 代表矩阵中的位置，$\hat{f}$ 代表量化的结果。

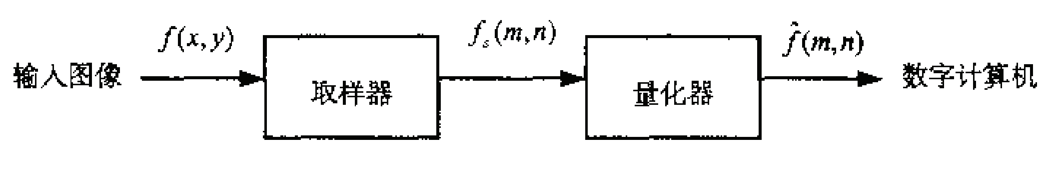


图 3.0-1 图像数字化的两大过程

3.1 取 样

取样是将图像 $f(x,y)$ 输入计算机的第一个处理过程。在取样过程中一个重要的考虑是 $f(x,y)$ 的取样密度应当多大，才不会漏失原图像的信息。不漏失信息而能完整地恢复原图像是对取样的基本要求。本节所讨论的取样定理将提供取样所应遵循的准则。

3.1-1 均匀矩形取样

设 $f(x,y)$ 为一个有限带宽 (bandlimited) 的二维连续函数。亦即其傅立叶变换 $F(u,v)=\mathcal{F}\{f(x,y)\}$ 有下列性质

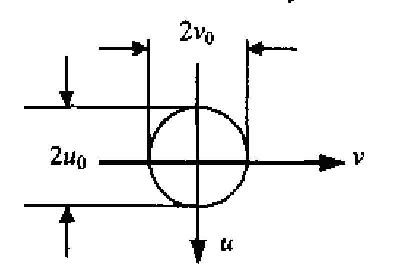
$$F(u,v)=0\,,\quad |u|>u_0\,,\quad |v|>v_0 \tag{3.1-1}$$

其中 u_0 和 v_0 分别表示在 u 和 v 方向上的截止频率。换言之，在傅立叶变换域中，原图像 $f(x,y)$ 的信息集中在长宽为 $2u_0$ 和 $2v_0$ 的矩形范围内的区域 R，如图 3.1-1 (a) 所示。

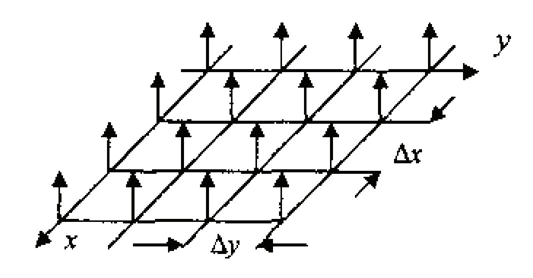
另外考虑一个二维取样函数

$$s(x,y)=\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}\delta(x-m\Delta x,y-n\Delta y) \tag{3.1-2}$$

它是沿 x 方向间隔为 Δx、沿 y 方向间隔为 Δy 的二维脉冲函数序列，如图 3.1-1 (b) 所示。




(a) 一个有限带宽函数在 uv 平面上的区域 R



(b) 取样函数 $S(x,y)$

图 3.1-1 (待续)

用取样函数 $s(x,y)$ 对 $f(x,y)$ 取样，得到取样图像 $f_s(x,y)$，此函数可以表示成强度为图像取样点灰度值的二维脉冲函数阵列，亦即

$$f_s(x,y)=s(x,y)f(x,y)=\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}f(x,y)\delta(x-m\Delta x,y-n\Delta y)$$
$$=\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}f(m\Delta x,n\Delta y)\delta(x-m\Delta x,y-n\Delta y) \tag{3.1-3}$$

这相当于是对于 $f(x,y)$ 以矩形点阵均匀取样，每个取样位置在 $x=m\Delta x$，$y=n\Delta y$ 上，其中 $m,n=0,\ \pm1,\ \pm2,\ \cdots$。

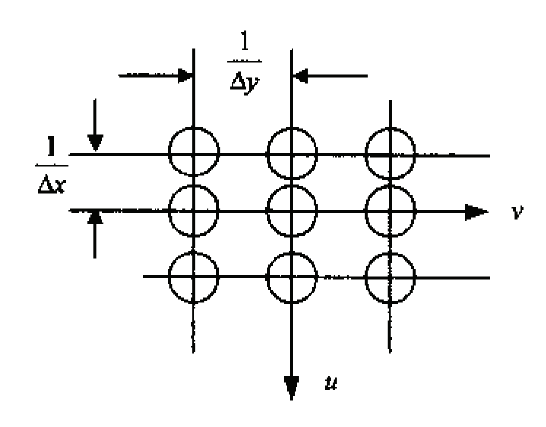
现在看 $f_s(x,y)$ 的频谱 $F_s(u,v)$。首先推导出 $s(x,y)$ 的傅立叶变换式 $S(u,v)$ 为

$$S(u,v)=\frac{1}{\Delta x}\frac{1}{\Delta y}\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}\delta\left(u-m\frac{1}{\Delta x},v-n\frac{1}{\Delta y}\right) \tag{3.1-4}$$

因为 $f_s(x,y)=s(x,y)f(x,y)$，故其傅立叶变换式 $F_s(u,v)$ 为 $S(u,v)$ 和 $F(u,v)$ 的卷积

$$F_s(u,v)=\mathcal{F}\{f_s(x,y)\}=\mathcal{F}\{s(x,y)f(x,y)\}=S(u,v)*F(u,v)$$
$$=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\frac{1}{\Delta x}\frac{1}{\Delta y}\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}\delta\left(\alpha-m\frac{1}{\Delta x},\beta-n\frac{1}{\Delta y}\right)F(u-\alpha,v-\beta)\mathrm{d}\alpha\mathrm{d}\beta$$
$$=\frac{1}{\Delta x}\frac{1}{\Delta y}\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\delta\left(\alpha-m\frac{1}{\Delta x},\beta-n\frac{1}{\Delta y}\right)F(u-\alpha,v-\beta)\mathrm{d}\alpha\mathrm{d}\beta$$
$$=\frac{1}{\Delta x}\frac{1}{\Delta y}\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}F\left(u-m\frac{1}{\Delta x},v-n\frac{1}{\Delta y}\right) \tag{3.1-5}$$

(3.1-5) 式显示取样后图像 $f_s(x,y)$ 的频谱 $F_s(u,v)$ 是原来 $f(x,y)$ 的频谱 $F(u,v)$ 分别沿 u 轴和 v 轴每隔 $\frac{1}{\Delta x}$ 和 $\frac{1}{\Delta y}$ 就复制一个 $F(u,v)$ 的结果，如图 3.1-1(c)所示。从图 3.1-1(c)中可以看出，当



(c) 取样图像的频谱

图 3.1-1　(续)

$f(x,y)$ 满足 (3.1-1) 式的有限带宽条件，且

$$\begin{cases} \dfrac{1}{\Delta x} \geqslant 2u_0 \\ \dfrac{1}{\Delta y} \geqslant 2v_0 \end{cases} \Rightarrow \begin{cases} \Delta x \leqslant \dfrac{1}{2u_0} \\ \Delta y \leqslant \dfrac{1}{2v_0} \end{cases} \tag{3.1-6}$$

时，各个相邻的区域 R 不会彼此混叠，因而可以用一个理想的低通滤波器取出一个完整的 R，使原信号 $f(x,y)$ 不失真地再现，这就是二维取样定理。

$f_s(x,y)$ 仅仅在 $x=m\Delta x$，$y=n\Delta y$ 的位置上有取样值，因此可以把 $f_s(x,y)$ 改写为 $f_s(m,n)$。$f_s(m,n)$ 形成离散的阵列，但其函数值仍为非离散的实数。

3.1-2 其他取样方法

上一节是采用空间位置上均匀取样使得取样格点为矩形或正方形的方法。此方法容易实现，对分析也方便。但是对于非矩形有限带宽的信号，采用非矩形格点取样通常可获得较佳的取样密度(取样点数/面积)。

● **取样矩阵**

考虑图 3.1-2 所示的取样格点，其中取样点位置可写成

$$\begin{bmatrix} x \\ y \end{bmatrix} = m\begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} + n\begin{bmatrix} 2\Delta x \\ -\Delta y \end{bmatrix} = m\boldsymbol{v}_0 + n\boldsymbol{v}_1 \tag{3.1-7}$$

其中 m 与 n 为任意整数，矩阵 $\boldsymbol{V}=[\boldsymbol{v}_0 \ \ \boldsymbol{v}_1]$ 称为取样矩阵。

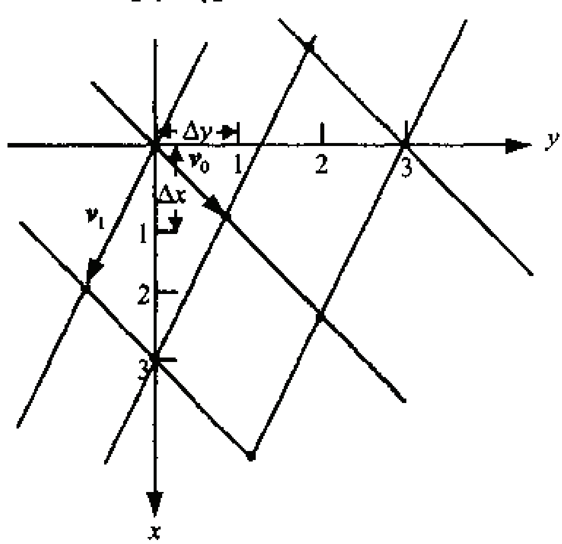


图 3.1-2 由取样矩阵 V 所产生的取样点阵

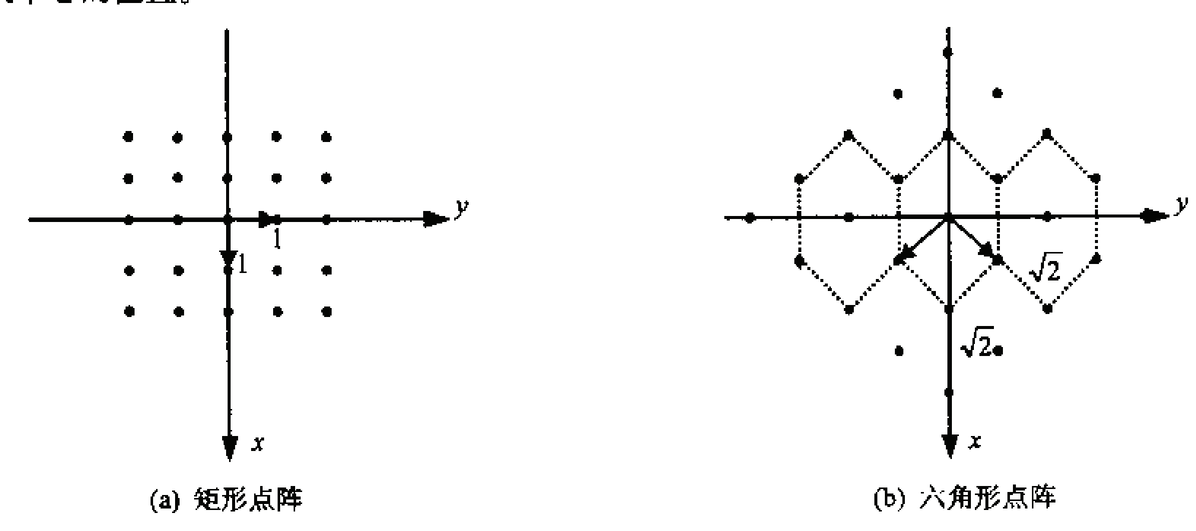
若取样矩阵为

$$\boldsymbol{V} = \begin{bmatrix} \Delta x & 0 \\ 0 & \Delta y \end{bmatrix} \tag{3.1-8}$$

则其取样点阵为如图 3.1-3(a)所示的矩形点阵；若取样矩阵为

$$\boldsymbol{V}=\begin{bmatrix}\frac{1}{\sqrt{2}}\Delta x & \frac{1}{\sqrt{2}}\Delta x\\ \frac{1}{\sqrt{2}}\Delta y & -\frac{1}{\sqrt{2}}\Delta y\end{bmatrix} \tag{3.1-9}$$

则其取样点阵为图 3.1-3 (b) 所示的非矩形点阵。此点阵中的点可视为置于某些六角形的顶点及中心的位置。



(a) 矩形点阵　　(b) 六角形点阵

图 3.1-3　不同取样矩阵所产生的两个不同点阵

对于某一个点阵，其所对应的取样矩阵$\boldsymbol{V}$不是唯一的；换言之，两个不同密度的取样矩阵可产生相同的点阵。

- **周期矩阵**

若存在一正数P，使得一个一维函数$F(u)$满足

$$F(u)=F(u-Pk)\,,\qquad \text{对所有}\,u \tag{3.1-10}$$

对所有整数k、则我们说此函数为周期性的。亦即我们将原点位置移到P的整数倍上、仍不改其函数。同理，对于N维函数$F(\Omega)$，若

$$F(\Omega)=F(\Omega-\boldsymbol{Uk})\,,\qquad \text{对所有}\,\Omega \tag{3.1-11}$$

对所有整数向量$\boldsymbol{k}$、则$F(\Omega)$为周期函数。其中矩阵$\boldsymbol{U}$称为周期矩阵。

(3.1-5) 式的$F_s(u,v)$为一周期函数，其周期矩阵为

$$\boldsymbol{U}=\begin{bmatrix}\frac{1}{\Delta x} & 0\\ 0 & \frac{1}{\Delta y}\end{bmatrix} \tag{3.1-12}$$

由 (3.1-8) 式及 (3.1-12) 式可看出

$$\boldsymbol{U}\boldsymbol{V}=\boldsymbol{I}_2 \tag{3.1-13}$$

其中 $\boldsymbol{I}_2$ 为 2×2 的单位矩阵。一般而言，$\boldsymbol{U}$ 与 $\boldsymbol{V}$ 有如下关系

$$\boldsymbol{U}=(\boldsymbol{V}^{-1})^{\mathrm{T}},\quad \boldsymbol{V}=(\boldsymbol{U}^{-1})^{\mathrm{T}} \tag{3.1-14}$$

● **取样密度**

考虑一个由 $\boldsymbol{v}_0$ 与 $\boldsymbol{v}_1$ 及其位移所构成的四边形 (如图 3.1-2 所示)。由于取样点数与四边形的个数一样，因此每单位面积的取样点数等于每单位面积所能放进去的四边形个数。因为 $|\det\boldsymbol{V}|$ ($\boldsymbol{V}$ 的行列式绝对值) 代表四边形面积，所以每单位面积上的格点数等于 $1/|\det\boldsymbol{V}|$，因此

$$\text{取样密度 } \rho=\frac{1}{|\det\boldsymbol{V}|} \tag{3.1-15}$$

已知一有限带宽函数的频宽范围和取样点阵模式，则有一使其取样不产生混叠的最小取样密度 $\rho_{\min}$ 与之对应。以图 3.1-1 (a) 的有限带宽范围以及图 3.1-1 (b) 的取样点阵模式为例，其取样矩阵如 (3.1-8) 式，故其周期矩阵如 (3.1-12) 式。因此其取样密度为

$$\rho=\frac{1}{|\det\boldsymbol{V}|}=\frac{1}{\Delta x\Delta y} \tag{3.1-16}$$

但由于 (3.1-6) 式的限制，即

$$\begin{aligned}\frac{1}{\Delta x}&\geqslant 2u_0\\ \frac{1}{\Delta y}&\geqslant 2v_0\end{aligned} \tag{3.1-17}$$

故其最小取样密度 $\rho_{\min}=4u_0v_0$。

对于 $u_0=v_0=r$ (即圆环形频谱范围) 的情况，我们考虑一种如图 3.1-4 所示的六角形取样方式。其周期矩阵为

$$\boldsymbol{U}=\begin{bmatrix}\sqrt{3}r & -\sqrt{3}r\\ r & r\end{bmatrix} \tag{3.1-18}$$

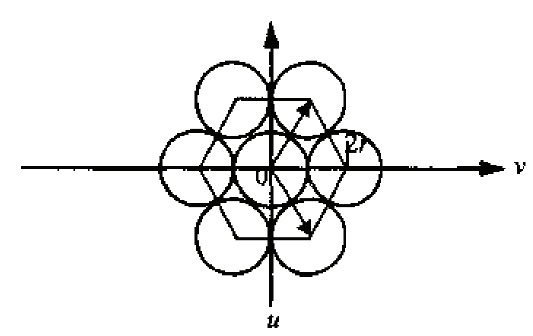


图 3.1-4　六角形取样

由 (3.1-14) 式可知其取样矩阵为

$$\boldsymbol{V}=(\boldsymbol{U}^{-1})^{\mathrm{T}}=\begin{bmatrix}\frac{1}{2\sqrt{3}}\frac{1}{r} & -\frac{1}{2\sqrt{3}}\frac{1}{r}\\ \frac{1}{2r} & \frac{1}{2r}\end{bmatrix} \tag{3.1-19}$$

故其最小取样密度

$$\rho_{\min}(\text{六角形})=\frac{1}{|\det \boldsymbol{V}|}=2\sqrt{3}r^2 \tag{3.1-20}$$

若采用矩形点阵，则其最小取样密度为

$$\rho_{\min}(\text{矩形})=4r^2 \tag{3.1-21}$$

因此

$$\frac{\rho_{\min}(\text{矩形})}{\rho_{\min}(\text{六角形})}=\frac{4r^2}{2\sqrt{3}r^2}\approx 1.15 \tag{3.1-22}$$

由此可知六角形取样比矩形取样更有效率 1.15 倍。另外可计算

$$\frac{\rho_{\min}(\text{矩形})-\rho_{\min}(\text{六角形})}{\rho_{\min}(\text{矩形})}=\frac{4r^2-2\sqrt{3}r^2}{4r^2}\approx 13.4\% \tag{3.1-23}$$

这表示对于圆环形有限带宽信号而言，采用六角形格点取样比采用矩形格点取样可减少 13.4%的取样点数。

3.1-3　重　建

图像的重建是从取样图像 $f_s(m,n)$ 还原到连续图像 $f(x,y)$ 的过程。此过程借助通过空间滤波器或者空间内插来完成。

用一个理想频率响应如下的二维低通滤波器

$$H(u,v)=\begin{cases}1, & |u|<u_0 \text{ 且 } |v|<v_0\\ 0, & \text{其他}\end{cases} \tag{3.1-24}$$

将 $H(u,v)$ 与 $F_s(u,v)$ 相乘，便可取出原来的 $F(u,v)$，经反变换后可以完整地得到 $f(x,y)$

$$F(u,v)=\kappa H(u,v)\cdot F_s(u,v) \tag{3.1-25}$$

即

$$f(x,y)=\kappa h(x,y)*f_s(x,y) \tag{3.1-26}$$

其中 κ 为比例因子，此处 $\kappa=\Delta x\Delta y$。

设原图像 $f(x,y)$ 的信息集中在长和宽分别为 $2u_0$ 和 $2v_0$ 的矩形范围内，且 $\Delta x=\frac{1}{2u_0}$，$\Delta y=\frac{1}{2v_0}$，则

$$\begin{aligned}\kappa h(x,y)&=\mathcal{F}^{-1}\{\kappa H(u,v)\}=\frac{\sin(2\pi u_0 x)}{2\pi u_0 x}\cdot\frac{\sin(2\pi v_0 y)}{2\pi v_0 y}\\&=\mathrm{sinc}(2\pi u_0 x)\cdot\mathrm{sinc}(2\pi v_0 y)\end{aligned}\tag{3.1-27}$$

把 (3.1-3) 和 (3.1-27) 式代入 (3.1-26) 式，则有

$$\begin{aligned}f(x,y)&=\sum_m\sum_n\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\mathrm{sinc}[2\pi u_0(x-\alpha)]\mathrm{sinc}[2\pi v_0(y-\beta)]\\&\quad\times f(m\Delta x,n\Delta y)\delta(\alpha-m\Delta x,\beta-n\Delta y)\mathrm{d}\alpha\,\mathrm{d}\beta\\&=\sum_m\sum_n\mathrm{sinc}[2\pi u_0(x-m\Delta x)]\times\mathrm{sinc}[2\pi v_0(y-n\Delta y)]f(m\Delta x,n\Delta y)\end{aligned}\tag{3.1-28}$$

上式显示重建图像是由位于 $x=m\Delta x$，$y=n\Delta y$ 上的许多个二维 sinc 函数加权求和的结果，而加权值就是取样图像的值。

由 sinc 函数进行图像重建可以得到理想的结果，但是 sinc 函数对应到一个理想的矩形滤波器，进行内插实际上并不可行。通常采用其他类型的内插来替代 sinc 函数。例如图 3.1-5 所示的几个函数，其中 (a) 是方形函数，可得到零阶样本内插；(b) 是三角函数，可得到一阶样本内插 (线性内插)；(c) 和 (d) 是由方形及三角形函数内插的结果。

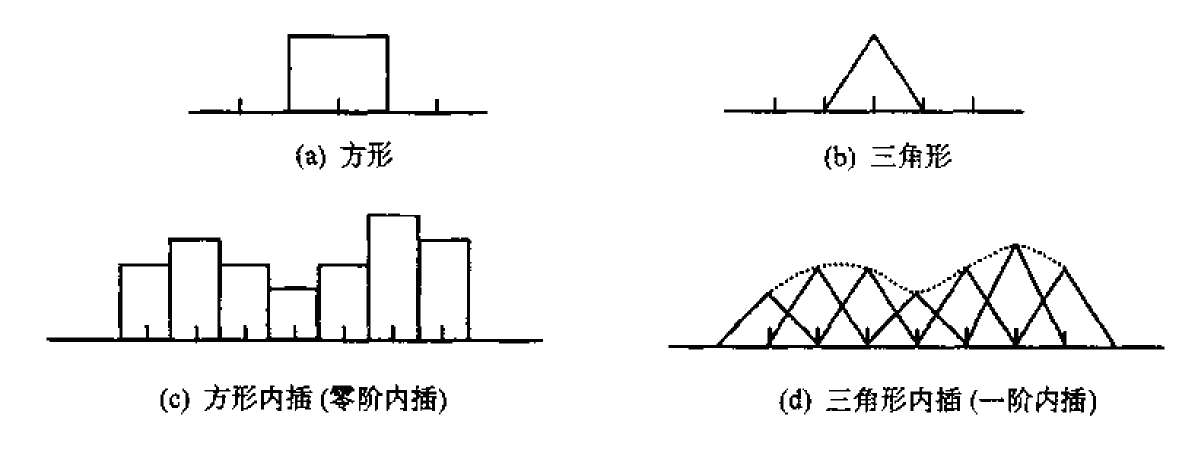
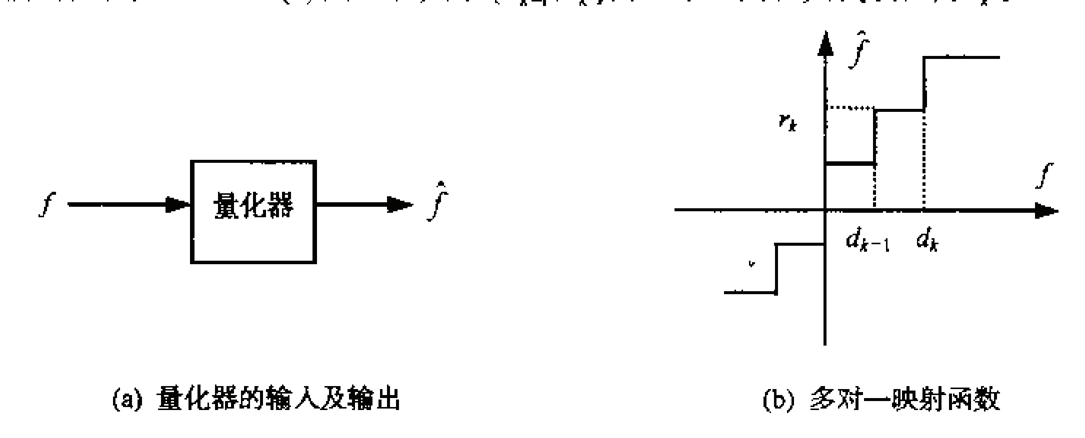


图 3.1-5 内插函数

3.2 量 化

如图 3.2-1 (a) 所示，对图像 $f(x,y)$ 取样后，得到取样值 $f=f_s(m,n)$。在进入计算机前，f_s 还需要进行量化。从数学的角度来看，所谓量化事实上是一个多对一函数的映射(mapping)关系。例如从一个有无穷多个可能数值的实数范围映射到有限个整数所构成的范围。从实现

的角度来看，就是把样本值的取值范围分成若干个区间，然后用某个代表值代表这一区间内所有可能的值，如图 3.2-1 (b) 所示，其中$(d_{k-1},d_k]$为一个区间，其代表值为r_k。



(a) 量化器的输入及输出　　(b) 多对一映射函数

图 3.2-1 将量化器视为数学的函数映射

为便于计算机的储存，一般常将取值范围分成2^B个区间，例如$B=6$、7、…、12，分别可以把像素的灰度值分成 64，128，…，4 096 个区间。区间越多，则由已量化的样本值(即计算机内的数字图像)恢复的实际图像越接近原图像，看上去越令人满意。当区间不够时，在图像的灰度值变化缓慢的区域内将出现原图像上没有的**伪轮廓** (false contour)。这是由于量化过于粗糙，使得量化噪声过大所致。这种伪轮廓有可能妨害我们从图像获取真正的信息，而令人无法接受。B愈大图像愈精致，也愈不容易有伪轮廓。但B也代表储存每个像素所需的位数，因此B愈大也代表计算机的储存量以及对其处理的计算量都大增。因为人眼的感官系统对图像精致程度的需求是有极限的，因此任意采取很大的B值是毫无必要的。以需求较高的医学图像而言，有人用$B=12$，但对一般图像而言$B=8$已足以应付大部分的应用。

最简单的量化方法就是把样本值的整个取值范围均匀地分成L个子区间，此即均匀量化的基本概念。若子区间的大小不一，则得到非均匀量化器。前者通常用于样本值大致均匀落在整个取值范围内时；后者对样本值在取值范围内出现机会不均等时有较佳的表现。所谓较佳是指在同等量化区间 (或同等位元) 下，非均匀量化实际引人的量化噪声较少。以下分别介绍这两种量化方式。

3.2-1 均匀量化

设取样后的结果为$f=f_s(m,n)$，$f\in(d_0,d_L]$。假设取样值在$(d_0,d_L]$的范围内有相同的出现机会，即其概率密度函数$p_f(f)=$常数P。现在把整个取值范围$(d_0,d_L]$均匀地分成L个子区间$(d_{k-1},d_k]$，$k=1,2,\cdots,L$。每个子区间$(d_{k-1},d_k]$对应到一个定值r_k，总共有L个定值r_k。其中的L个r_k，$1\leqslant k\leqslant L$，称为**重构** (reconstruction)层；另外还有 $(L+1)$ 个d_k，$0\leqslant k\leqslant L$，称为**决策** (decision) 边界。

整个量化的规则很简单：当$f\in(d_{k-1},d_k]$时，对应到量化值

$$\hat{f}=r_k \tag{3.2-1}$$

可写成

$$\hat{f}=Q(f)=r_k, d_{k-1}<f\leqslant d_k \tag{3.2-2}$$

此量化造成的误差为

$$e_Q=\hat{f}-f \tag{3.2-3}$$

有许多 f 与 $\hat{f}$ 之间的失真测度方式 $d(f,\hat{f})$，例如

$$\begin{aligned} d(f,\hat{f})&=\left|\hat{f}-f\right| \\ d(f,\hat{f})&=\left|\left|\hat{f}\right|^N-\left|f\right|^N\right| \end{aligned} \tag{3.2-4}$$

其中 N 为正整数。但最常用的却是

$$e_Q^2=\left(\hat{f}-f\right)^2 \tag{3.2-5}$$

量化器的最优化设计就是要选取 d_k 与 r_k 使某个 f 与 $\hat{f}$ 之间的失真测度最小化，例如我们可选择使平均失真 D 最小

$$D=E\left[d(f,\hat{f})\right]=\int_{-\infty}^{\infty}d(f_0,\hat{f})p_f(f_0)\mathrm{d}f_0 \tag{3.2-6}$$

现在分析采用 e_Q^2 失真测度时均匀量化所造成的误差。当 $f\in(d_{k-1},d_k]$ 时，取样值量化为 $\hat{f}=r_k$，因而有误差 (r_k-f)。在 $f\in(d_{i-1},d_i]$ 区间取样值为 f 的概率密度是 $p_f(f)$，因此在该区间内所造成误差平方的统计平均为

$$\int_{d_{i-1}}^{d_i}(\hat{f}-f_0)^2p_f(f_0)\mathrm{d}f_0 \tag{3.2-7}$$

因为 $(d_0,d_L]$ 范围内的概率密度 $p_f(f)=$ 常数 P，因此 L 个子区间误差平方的总和 D 为

$$\begin{aligned} D&=\sum_{i=1}^{L}\int_{d_{i-1}}^{d_i}(\hat{f}-f_0)^2p_f(f_0)\mathrm{d}f_0=P\sum_{i=1}^{L}\int_{d_{i-1}}^{d_i}(r_i-f_0)^2\mathrm{d}f_0 \\ &=\frac{1}{3}P\sum_{i=1}^{L}[(r_i-d_{i-1})^3-(r_i-d_i)^3] \end{aligned} \tag{3.2-8}$$

要取得 D 的最小值，故取 $\frac{\partial D}{\partial r_k}=0$，即

$$(r_k-d_{k-1})^2-(r_k-d_k)^2=0,\quad 1\leqslant k\leqslant L \tag{3.2-9}$$

由此得到最佳量化值

$$r_k=\frac{1}{2}(d_k+d_{k-1}) \tag{3.2-10}$$

(3.2-10) 式显示若取样值 f 在 $(d_0,d_L]$ 内为均匀分布，则量化值 r_k 取每个子区间 $(d_{k-1},d_k]$ 的中间值可得最小的量化误差。

设子区间 $(d_{k-1},d_k]$ 的长度为 Δ，则 (3.2-8) 式中 $P=\dfrac{1}{L\Delta}$，此时我们有

$$r_k-d_k=-\frac{\Delta}{2},\quad r_k-d_{k-1}=\frac{\Delta}{2} \tag{3.2-11}$$

将此式代入 (3.2-8) 式中得

$$D=\frac{1}{3}\frac{1}{L\Delta}\sum_{i=1}^{L}\left[\left(\frac{\Delta}{2}\right)^3-\left(-\frac{\Delta}{2}\right)^3\right]=\frac{\Delta^2}{12} \tag{3.2-12}$$

由此可见，当量化区间数 L 加大时，Δ 成比例地缩小，D 会以成平方反比大幅缩小。因此加大量化区间的数目 L 对原图像的保真度有很大的帮助。

3.2-2　非均匀量化

当概率密度 $p_f(f)$ 不再是常数时，非均匀量化是较佳的选择。由于子区间越大，造成的量化误差就越大，因此在概率密度 $p_f(f)$ 较小处，可取较大的量化区间长度；反之，则取较小的量化区间，这就是非均匀量化的基本概念。以下推导采用 e_Q^2 失真测度的非均匀量化器的 d_k 与 r_k。

同样由 (3.2-7) 式开始，

$$\begin{aligned} D&=\sum_{i=1}^{L}\int_{d_{i-1}}^{d_i}(\hat{f}-f_0)^2p_f(f_0)\mathrm{d}f_0 \\ &=\cdots+\int_{d_{k-1}}^{d_k}(r_k-f_0)^2p_f(f_0)\mathrm{d}f_0+\int_{d_k}^{d_{k+1}}(r_{k+1}-f_0)^2p_f(f_0)\mathrm{d}f_0+\cdots \end{aligned} \tag{3.2-13}$$

欲得 (3.2-13) 式中 D 的最小值，故求 D 对 d_k 和 r_k 的偏导数，并令其为 0，即

$$\frac{\partial D}{\partial d_k}=(r_k-d_k)^2p_f(d_k)-(r_{k+1}-d_k)^2p_f(d_k)=0 \tag{3.2-14}$$

其中 $k=1,2,\cdots,L-1$，一共有 $L-1$ 个式子，此处 d_0 和 d_L 是已知的，不必对它们再求偏导数。另外令

$$\frac{\partial D}{\partial r_k}=2\int_{d_{i-1}}^{d_i}(r_k-f_0)p_f(f_0)\mathrm{d}f_0=0 \tag{3.2-15}$$

其中 $k=1,2,\cdots,L$，一共有 L 个式子。

若 $p_f(d_k)\neq 0$，则由 (3.2-14) 式可得

$$d_k=\frac{1}{2}(r_k+r_{k+1}),\qquad k=1,2,\cdots,L-1 \tag{3.2-16}$$

从 (3.2-15) 式可推导出

$$r_k=\frac{\int_{d_{k-1}}^{d_k} f_0 p_f(f_0)\mathrm{d}f_0}{\int_{d_{k-1}}^{d_k} p_f(f_0)\mathrm{d}f_0}=E\{f|f\in(d_{k-1},d_k]\},\qquad k=1,2,\cdots,L \tag{3.2-17}$$

由 (3.2-16) 式及 (3.2-17) 式可知，最佳量化器的各子区间决策边界 d_k 应当是量化重构层准值 r_k 间的中间值，且每一个 r_k 是 f 落在子区间 $(d_{k-1},d_k]$ 的条件下的条件期望值。(3.2-16) 与 (3.2-17) 式彼此有牵连，类似鸡生蛋、蛋生鸡的问题，因此必须采用计算机的数值解法求出 d_k 和 r_k。给定 d_0、d_L、$p_f(f)$、L 之后，求 d_k 及 r_k 的计算机解法如下：

(1) 先假定一个 r_1 的值，由 (3.2-17) 式取 $k=1$，求出 d_1。由于 d_1 是积分式的上限，故要采用数值近似计算，逐步逼近 d_1 的解；

(2) 已知 d_1 和 r_1，在 (3.2-16) 式中取 $k=1$，求出 r_2；

(3) 由 d_1 和 r_2，在 (3.2-17) 式中，取 $k=2$，用数值近似计算逐步逼近 d_2 的解；

(4) 由 d_2 和 r_2，在 (3.2-16) 式中，取 $k=2$，求出 r_3。

(5) 其他依此类推，最终求出 r_L，代入 (3.2-17) 式，看其是否等于 $\frac{\int_{d_{L-1}}^{d_L} f_0 p_f(f_0)\mathrm{d}f_0}{\int_{d_{L-1}}^{d_L} p_f(f_0)\mathrm{d}f_0}$。如果不相等，则重新假定 r_1 值，再回到第 (1) 步，继续计算求出 r_L，直到 r_L 符合要求为止。

由以上程序所得的量化器称为最佳最小均方或 Lloyd-Max 量化器，其中与图像处理最相关的概率密度函数为

$$\text{高斯：}\ p_f(f)=\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(\frac{-(f-\mu)^2}{2\sigma^2}\right) \tag{3.2-18}$$

$$\text{拉普拉斯：}\ p_f(f)=\frac{\alpha}{2}\exp\left(-\alpha|f-\mu|\right) \tag{3.2-19}$$

其中 μ 与 σ^2 分别代表 f 的平均与方差，拉普拉斯概率密度函数的方差为 $\sigma^2=2/\alpha$。

最佳均方误差量化器有几个性质：

1. 量化器的输出是输入的不偏 (unbiased) 估测，即

$$E[\hat{f}]=E[f] \tag{3.2-20}$$

2. 量化误差对量化器的输出为正交，亦即

$$E[(\hat{f}-f)\hat{f}]=0 \tag{3.2-21}$$

3. 若 d_k 与 r_k 分别为零平均且方差为 1 的随机变量 f 的判决层与重构层，则

$$\tilde{d}_k=\mu+\sigma d_k,\ \tilde{r}_k=\mu+\sigma r_k \tag{3.2-22}$$

分别为与f有相同分布但平均为μ且方差为σ^2的随机变量所需要的判决层与重构层。以上性质的证明当作习题。

如果$p_f(f)$有如下的均匀分布

$$p_f(f)=\begin{cases}\dfrac{1}{d_L-d_0}, & d_0\leqslant f\leqslant d_L\\ 0, & \text{其他情况况}\end{cases} \tag{3.2-23}$$

则由 (3.2-17) 式可得

$$r_k=\frac{\frac{1}{2}(d_k{}^2-d_{k-1}{}^2)}{d_k-d_{k-1}}=\frac{1}{2}(d_k+d_{k-1}) \tag{3.2-24}$$

又由 (3.2-16) 式，我们有

$$d_k=\frac{1}{2}(r_k+r_{k+1}) \tag{3.2-25}$$

结合 (3.2-24) 与 (3.2-25) 两式可得

$$d_k=\frac{1}{2}\left[\frac{1}{2}(d_k+d_{k-1})+\frac{1}{2}(d_{k+1}+d_k)\right] \tag{3.2-26}$$

化简上式得到

$$d_k=\frac{1}{2}(d_{k-1}+d_{k+1}) \tag{3.2-27}$$

由 (3.2-27) 式可推出

$$d_k-d_{k-1}=d_{k+1}-d_k=\varDelta \quad (\text{常数}) \tag{3.2-28}$$

同理由 (3.2-24) 式亦可推得

$$r_{k+1}-r_k=\frac{1}{2}(d_{k+1}+d_k)-\frac{1}{2}(d_k+d_{k-1})=\frac{1}{2}(d_{k+1}-d_{k-1})=\varDelta \quad (\text{常数}) \tag{3.2-29}$$

以上说明当输入值有均匀概率密度函数时，非均匀量化已蜕变为均匀量化，因此均匀量化可视为非均匀量化的一个特例。

对于上述均匀量化器，其误差

$$e_Q=\hat{f}-f \tag{3.2-30}$$

均匀分布在$-\dfrac{\varDelta}{2}\sim\dfrac{\varDelta}{2}$之间，因此其均方误差为

$$D=E\left[e_Q{}^2\right]=\frac{1}{\varDelta}\int_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}}e_Q{}^2\mathrm{d}e_Q=\frac{\Delta^2}{12} \tag{3.2-31}$$

正如所料，与 (3.2-12) 式的结果一样。

我们知道对一个均匀分布的随机变量，若其范围为 A 时其方差 $\sigma^2=\dfrac{A^2}{12}$。对于有 B 位的均匀量化器，我们有 $\Delta=\dfrac{A}{2^B}$。因此其信噪比 (SNR) 为

$$\frac{\sigma^2}{D}=\frac{\frac{A^2}{12}}{\frac{\Delta^2}{12}}=\left(\frac{A}{\Delta}\right)^2=2^{2B} \tag{3.2-32}$$

以分贝 (dB) 计，则

$$\mathrm{SNR}=10\log_{10}2^{2B}\approx 6B\,(\mathrm{dB}) \tag{3.2-33}$$

因此对于均匀分布的最佳均方误差量化器所能获得的信噪比是每一位约有 6 dB 的增益。

3.2-3　压缩扩展型量化器

一个压缩扩展器 (compander) 是由一个称为压缩器 (compressor)，加上一个均匀量化器，以及一个扩展器 (expander) 所形成，如图 3.2-2 所示。所谓压缩与扩展是指对输出入信号的动态范围而言。整个压缩扩展器具有非均匀量化的功能，此功能来自于压缩器与扩展器这两个非线性变换函数。

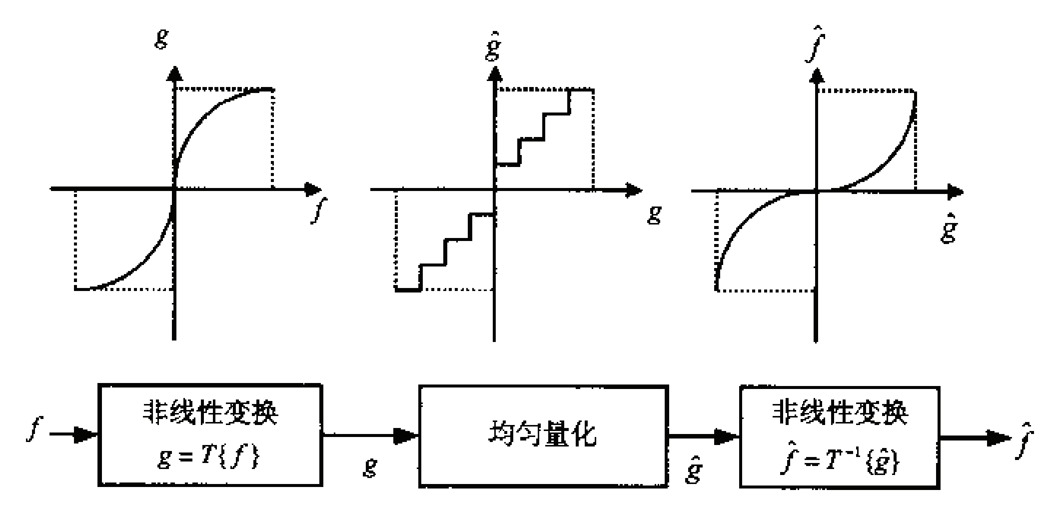


图 3.2-2　一个压缩扩展型量化器

在压缩扩展型量化器中，首先做一非线性变换，即

$$g=T\{f\} \tag{3.2-34}$$

使变换后的结果 g 的概率密度函数 $p_g(g)$ 接近均匀的，然后进行 3.2-1 节所讨论的均匀量化。由於均匀量化器对有均匀密度函数的输入而言为最佳，因此压缩器应使非均匀概率分布的输入尽可能变成均匀分布的输出。最后扩展器再执行另一个非线性的逆变换函数，即

$$f = T^{-1}\{g\} \tag{3.2-35}$$

最后得到量化输出 $\hat{f}$。

若

$$p_g(g) = \begin{cases} 1, & -1/2 \leqslant g \leqslant 1/2 \\ 0, & \text{其他情况} \end{cases} \tag{3.2-36}$$

且若 f 为一零平均的随机变量，则一个适当的变换为

$$g = T\{f\} = \int_{-\infty}^{f} p_f(f_0)\,\mathrm{d}f_0 - 1/2 \tag{3.2-37}$$

亦即非线性变换函数相当于 f 的累积概率分布。例如对雷利 (Rayleigh) 概率密度函数

$$p_f(f) = \frac{f}{\sigma^2}\exp\left(-\frac{f^2}{2\sigma^2}\right) \tag{3.2-38}$$

我们可求得其正变换为

$$g = \frac{1}{2} - \exp\left(-\frac{f^2}{2\sigma^2}\right) \tag{3.2-39}$$

而其反变换为

$$f = \left\{2\sigma^2 \ln\left[1\Big/\left(\frac{1}{2} - g\right)\right]\right\}^{\frac{1}{2}} \tag{3.2-40}$$

3.3　向量量化

对所谓无记忆 (memoryless) 的信息源而言，我们是把信号的各个取样值都视为互不相关彼此独立，此时上两节所讨论的均匀量化与非均匀量化对个别取样点逐点量化的过程是合理的做法。但是大多数实际信号各取样值之间存在有相关性，亦即知道某个取样值的参数，对其邻近取样值亦可做合理推测。由此观念所衍生的量化方法称为向量量化 (vector quantization,VQ)，亦即把若干取样值集合成一个向量，以此向量为量化的单位而不是以一个取样点为单位。相对于 VQ，前二节所述逐点量化的过程称为纯量量化 (scalar quantization, SQ)。

设 $f = [f_1, f_2, \cdots f_N]^{\mathrm{T}}$ 表示一 N 维向量，它是由 N 个实数连续纯量值 f_i 所组成。在 VQ 中，$\boldsymbol{f}$ 将被映射到另一个 N 维向量 $\boldsymbol{r} = [r_1, r_2, \cdots, r_N]^{\mathrm{T}}$。$\boldsymbol{f}$ 的 VQ 是将一个 N 维向量空间分割成 L 个决策区域 C_i，$1 \leqslant i \leqslant L$，每一个决策区域包围一个重建向量 $\boldsymbol{r}_i$。在编码的文献中常将 $\boldsymbol{r}_i$ 称为码向量 (codevector)，并将码向量所形成的集合称为码本 (codebook)。

设 $\hat{\boldsymbol{f}}$ 代表 $\boldsymbol{f}$ 量化的结果，则可写成

$$\hat{\boldsymbol{f}} = \mathrm{VQ}(\boldsymbol{f}) = \boldsymbol{r}_i, \quad \boldsymbol{f} \in C_i \tag{3.3-1}$$

其中 VQ 代表向量量化的操作。

与 SQ 的情形一样，我们可定义一个失真测度$d(\boldsymbol{f},\hat{\boldsymbol{f}})$。一个常用的$d(\boldsymbol{f},\hat{\boldsymbol{f}})$是$\boldsymbol{e}_Q^{\mathrm{T}}\boldsymbol{e}_Q$，其中量化误差$\boldsymbol{e}_Q$定义为

$$\boldsymbol{e}_Q=\hat{\boldsymbol{f}}-\boldsymbol{f} \tag{3.3-2}$$

重建向量$\boldsymbol{r}_i$以及决策区域C_i的边界可由使某个失真量最小化来决定，例如采用平均失真$D=E[d(\boldsymbol{f},\hat{\boldsymbol{f}})]$。若$d(\boldsymbol{f},\hat{\boldsymbol{f}})$为$\boldsymbol{e}_Q^{\mathrm{T}}\boldsymbol{e}_Q$，则由 (3.3-1) 及 (3.3-2) 两式可得

$$\begin{aligned}D&=E[\boldsymbol{e}_Q^{\mathrm{T}}\boldsymbol{e}_Q]\\&=E[(\hat{\boldsymbol{f}}-\boldsymbol{f})^{\mathrm{T}}\cdot(\hat{\boldsymbol{f}}-\boldsymbol{f})]\\&=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty}(\hat{\boldsymbol{f}}-\boldsymbol{f}_0)^{\mathrm{T}}(\hat{\boldsymbol{f}}-\boldsymbol{f}_0)p_f(\boldsymbol{f}_0)\mathrm{d}\boldsymbol{f}_0\\&=\sum_{i=1}^{L}\int_{C_i}(\boldsymbol{r}_i-\boldsymbol{f}_0)^{\mathrm{T}}(\boldsymbol{r}_i-\boldsymbol{f}_0)p_f(\boldsymbol{f}_0)\,\mathrm{d}\boldsymbol{f}_0\end{aligned} \tag{3.3-3}$$

取$\dfrac{\partial D}{\partial \boldsymbol{r}_i}=0$可得

$$\int_C(\boldsymbol{r}_i-\boldsymbol{f}_0)p_f(\boldsymbol{f}_0)\mathrm{d}\boldsymbol{f}_0=0 \tag{3.3-4}$$

将上式重新整理后得

$$\begin{aligned}\boldsymbol{r}_i&=\frac{\int_C \boldsymbol{f}\,p_f(\boldsymbol{f}_0)\mathrm{d}\boldsymbol{f}_0}{\int_C p_f(\boldsymbol{f}_0)\mathrm{d}\boldsymbol{f}_0}\\&=E\{\boldsymbol{f}\mid\boldsymbol{f}\in C_i\}\end{aligned} \tag{3.3-5}$$

对于一个固定决策区间的分析，要决定最佳的重建向量$\boldsymbol{r}_i$必须知道联合概率密度$p_f(\boldsymbol{f})$。但是这个信息在实际问题中往往并不可知。此外，计算 (3.3-5) 式求$\boldsymbol{r}_i$也有实际上的困难，此困难随着向量维度的增加而加大。

基本上设计最佳$\boldsymbol{r}_i$所形成的码本是一个高度非线性的问题。一般而言采取的解决方案是利用以下两个必要条件

1. 最近距离条件

输入讯号向量$\boldsymbol{f}$，使其对应的代表向量$\boldsymbol{r}_i$的选择应满足

$$d(\boldsymbol{f},\boldsymbol{r}_i)\leqslant d(\boldsymbol{f},\boldsymbol{r}_j)\qquad \forall j\neq i \tag{3.3-6}$$

则称此不等式为使平均失真度测度 D 为最小的最近距离条件。

2. 中心条件

每个重建向量$\boldsymbol{r}_i$必须使C_i中的平均失真最小，亦即将

$$E[d(\boldsymbol{f},\boldsymbol{r}_i)\mid\boldsymbol{f}\in C_i] \tag{3.3-7}$$

对 $\boldsymbol{r}_i$ 最小化。满足 (3.3-7) 式的向量 $\boldsymbol{r}_i$ 称为 C_i 的重心 (centroid)。

以上两个条件可引出一个设计最佳码本的迭代过程。首先给一个 $\boldsymbol{r}_i$ 的初估值，理论上可将所有可能的 $\boldsymbol{f}$ 代入 (3.3-6) 式的条件中找到 C_i 的估测。给了 C_i 的估测后，计算 (3.3-7) 式的条件中的重心以得 $\boldsymbol{r}_i$，以此 $\boldsymbol{r}_i$ 作为一新估测值重复上述操作。这个过程有两个实际的问题：首先，需要所有可能的 $\boldsymbol{f}$ 以决定 C_i；其次，计算 C_i 的重心所需的 $p_f(\boldsymbol{f})$ 一般也不可得。实际上只有某些与我们要做向量量化近似的所谓"训练向量"，因此有人将上述的迭代程序修改成 K-平均 (K-mean) 算法，此处 K 等于码本大小或代表向量 $\boldsymbol{r}_i$ 的个数 L。

K-平均法的纯量版本 (向量维度为一) 由 Lloyd 于 1957 年提出，Forgy 在 1965 年提到变成一般向量量化的状况(Forgy [1965])。此方法又称 LBG 算法，这是因为 Linde、Buzo 与 Gray 三人在 1980 年所发表的一篇论文中详论 VQ 码本的设计(Linde et al. [1980])。以下是 LBG 算法的步骤。

(1) 初始化：定出码本大小 L、失真临界值 ε、初始码本 $R_0=\{\boldsymbol{r}_{i,0};\ 1\leqq i\leqq L\}$ 及训练序列 $T=\{\boldsymbol{f}_n;\ n=1,2,\cdots,N\}$，$N>>L$。设迭代次数 $m=0$，初始失真 $D_{-1}=\infty$。

(2) 对码本 $R_m=\{\boldsymbol{r}_{i,m};\ 1\leqq i\leqq L\}$，找出训练序列 T 的最小误差分割，即若

$$d(\boldsymbol{f}_n,\boldsymbol{r}_{i,m})\leqq d(\boldsymbol{f}_n,\boldsymbol{r}_{j,m}),\qquad \forall j\neq i$$

则 $\boldsymbol{f}_n\in C_i$。

(3) 计算平均失真

$$D_m=\frac{1}{N}\sum_{n=1}^{N}\min_{1\leqslant i\leqslant L}d(\boldsymbol{f}_n,\boldsymbol{r}_{i,m})$$

若 $(D_{m-1}-D_m)/D_m\leqslant\varepsilon$，则输出码本 R_m，退出迭代过程；否则继续。

(4) 设属于 C_i 的向量有 M_i 个，则新估测的 $\boldsymbol{r}_i$ 是使

$$\frac{1}{M_i}\sum_{\boldsymbol{f}\in C_i}d(\boldsymbol{f},\boldsymbol{r}_i)$$

最小的值。若 $d(\boldsymbol{f},\boldsymbol{r}_i)=(\boldsymbol{f}-\boldsymbol{r}_i)^{\mathrm{T}}(\boldsymbol{f}-\boldsymbol{r}_i)$，即平方误差，则 $\boldsymbol{r}_i$ 正是 M_i 个向量的算术平均值。

(5) 取 $m=m+1$，回到步骤(2)。

以上过程用 Matlab 程序写成，列于第三章实习的程序范例中。除了 LBG 算法之外，还有许多 VQ 的设计方法，例如人工神经网络中的自组织 (self-organization) 映射以及模拟退火 (simulated annealing) 等。

有关 VQ 的研究目前朝几个方向在进行：

① 训练序列与长度的选择。

② 失真测度的选择。

③ 改进最佳分割迭代的收敛速度。

④ 缩短最佳向量的比对时间。

⑤ 减少码本大小。

⑥ 发展自适性码本以扩大适用性。

⑦ 发展多级残余向量量化。

⑧ 与其他方法结合：例如与第四章中讨论到的小波变换。

1996 年 2 月的 IEEE Transactions on Image Processing 是专门讨论 VQ 的专刊，值得参考。

习　题

1. 设一图像信号为 $f(x,y)=4\cos 4\pi x\cos 6\pi y$，以 $\Delta x=\Delta y=0.5$ 与 $\Delta x=\Delta y=0.2$ 分别做取样。采用的重建滤波器为一截止频率 $\left(\frac{1}{2}\Delta x,\ \frac{1}{2}\Delta y\right)$ 为的理想低通滤波器，试问两种情况下的重建图像分别为何？
2. 将一个以 8 位表示每一个像素的灰度图像分别改以 2、4、6 位元均匀量化器输出图像并比较其结果。
3. 设一图像函数 $f(m,n)=255\mathrm{e}^{-\left[(m-m_0)^2+(n-n_0)^2\right]}$，其中 m_0 与 n_0 为固定常数。若将此函数以 a 位将其数字化并假设当两相邻像素的灰度值在 8 以上时人眼能感受到其差异。试问 a 取多少时会感受到伪轮廓？
4. 证明 3.2 节所列最佳均方误差量化器的三个性质。
5. 假设我们要用固定位数 B 对纯量量化后的 N 个纯量编码，其中第 i 个纯量 f_i 分配到的位数为 B_i，即 $B=\sum_{i=1}^{N}B_i$，其最佳位配置的策略取决于所用的失真测度及纯量的概率密度函数。通常对于协方差较大的纯量会分配较多之位数，反之则获得较少位数。假设除方差外，所有纯量的概率密度函数都一样，且都采用如 Lloyd-Max 等同一种量化器，则一个位数配置的近似解为

$$B_i=\frac{B}{N}+\frac{1}{2}\log_2\frac{\sigma_i^2}{\left[\prod_{j=1}^{N}\sigma_j^2\right]^{\frac{1}{N}}},\qquad 1\leqslant i\leqslant N$$

其中 σ_i^2 为 f_i 的方差。试计算每个纯量 f_i 的重构层个数 L_i。此个数与 σ_i 有何关系？
6. 向量量化较纯量量化的表现为佳是因为前者利用纯量间彼此的统计相关性以及维度大小，试举例分别说明这两种情况。
7. 修改第三章实习 L3.1 节所附的程序范例，对均匀、拉普拉斯及雷利概率密度函数的情况，重做该范例。
8. 将 LBG 算法用在一图像上，以获得向量维度为 16，且码本大小为 256 码本。显示量化前(即原图像)后两图像，评述其图像品质的差别。

第四章

变 换 法



由于图像上的某些问题在原函数定义的领域内较不易甚至是无法解决，所以借助某一个变换法将其变换到另一领域，使得问题较易处理。一般而言，我们希望变换具有以下三种特性：

1. 图像内像素间的相关性要降低

也就是说将最大部分的能量集中在最少数的变换系数上，使得像素间的信息冗余性能够去除。

2. 与图像无关的基底函数

不同的图像其统计特性通常也不同，而所谓最佳的变换与图像的统计特性有关，因此最佳变换通常随着图像的不同而有异。由于要找出最佳变换的基底函数非常耗时，尤其是当图像方块本身就属非平稳的时候。每个方块各有自己需要的基底函数以达到去除其相关性的目的，因此需要大量的计算。此外若以变换法做数据压缩（见第七章)、则各个方块的基底函数必须送至解码端，数据压缩效率会降低很多。因此，通常会舍弃最优化的变换而采用与图像无关的基底函数。

3. 快速完成变换

对于 N 点的变换，所需的计算量一般为 $O(N^2)$的等级，有些变换有较快速的算法能减少计算量，使其降为 $O(N\log_2 N)$ 的等级，因此对一个 $N\times N$ 的二维变换，若采取依序做列再做行的一维变换方式，其所需的计算量约为 $O(N^2\log_2 N)$ 而非 $O(N^4)$ 的等级。

4.1 正交变换

由一般序列的正交变换可直接延伸到一个对 $N\times N$ 图像 $f(m,n)$ 的正交变换

$$F(k,l)=\sum_{m=0}^{N-1}\sum_{n=0}^{N-1} f(m,n)\cdot a_{k,l}(m,n),\quad 0\leqslant k,l\leqslant N-1 \tag{4.1-1}$$

$$f(m,n)=\sum_{k=0}^{N-1}\sum_{l=0}^{N-1} F(k,l)\cdot a_{k,l}^{*}(m,n),\quad 0\leqslant m,n\leqslant N-1 \tag{4.1-2}$$

其中$\left\{a_{k,l}(m,n)\right\}$称为图像变换基底，是完整正交离散基底函数的集合，此集合满足以下性质：

1. 规范正交性 (Orthonormality)

$$\sum_{m=0}^{N-1}\sum_{n=0}^{N-1} a_{k,l}(m,n)\cdot a_{k',l'}^{*}(m,n)=\delta(k-k',l-l') \tag{4.1-3}$$

2. 完备性 (Completeness)

$$\sum_{k=0}^{N-1}\sum_{l=0}^{N-1} a_{k,l}(m,n)\cdot a_{k,l}^{*}(m',n')=\delta(m-m',n-n') \tag{4.1-4}$$

元素 $F(k,l)$ 称做变换系数，而$\left\{F(k,l)\right\}$称为变换图像，其规范正交性质保证任何被截短的部分和

$$f_{P,Q}(m,n) \equiv \sum_{k=0}^{P-1}\sum_{l=0}^{Q-1} F(k,l)a_{k,l}^*(m,n), \qquad P \leqslant N,\ Q \leqslant N \tag{4.1-5}$$

的下列平方误差

$$\sigma_e^2 = \sum_{m=0}^{N-1}\sum_{n=0}^{N-1}[f(m,n) - f_{P,Q}(m,n)]^2 \tag{4.1-6}$$

在系数 $F(k,l)$ 如 (4.1-1) 式所示时为最小，且当 $P=Q=N$ 时，该性质保证上述最小平方误差为零。

● **可分离么正变换** (Separable Unitary Transforms)

一个可分离变换的条件为

$$a_{k,l}(m,n) = a_k(m)b_l(n) \equiv a(k,m)b(l,n) \tag{4.1-7}$$

其中 $\{a_k(m), k=0,\cdots,N-1\}$，$\{b_l(n), l=0,\cdots,N-1\}$ 是一维基底向量所组成的正交集合。并以 $\boldsymbol{A} \equiv \{a(k,m)\}$ 和 $\boldsymbol{B} \equiv \{b(l,n)\}$ 表示，则由 (4.1-3) 及 (4.1-4) 式可知它们本身必为么正矩阵，亦即

$$\boldsymbol{A}\boldsymbol{A}^{*\mathrm{T}} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}^* = \boldsymbol{I}, \quad \boldsymbol{B}\boldsymbol{B}^{*\mathrm{T}} = \boldsymbol{B}^{\mathrm{T}}\boldsymbol{B}^* = \boldsymbol{I} \tag{4.1-8}$$

通常 $\boldsymbol{B}$ 和 $\boldsymbol{A}$ 相同，所以 (4.1-1) 和 (4.1-2) 式分别变成

$$F(k,l) = \sum_{m=0}^{N-1}\sum_{n=0}^{N-1} a(k,m)f(m,n)a(l,n) \leftrightarrow \boldsymbol{F} = \boldsymbol{A}\boldsymbol{f}\boldsymbol{A}^{\mathrm{T}} \tag{4.1-9}$$

$$f(m,n) = \sum_{k=0}^{N-1}\sum_{l=0}^{N-1} a^*(k,m)F(k,l)a^*(l,n) \leftrightarrow \boldsymbol{f} = \boldsymbol{A}^{*\mathrm{T}}\boldsymbol{F}\boldsymbol{A}^* \tag{4.1-10}$$

对于一 $N\times N$ 大小的图像，其变换系数 $F(k,l)$，使用 (4.1-1) 式所需的加法和乘法约为 $O(N^4)$，这样的计算量实在太大。对可分离变换而言，由 (4.1-9) 式中矩阵相乘的计算复杂度可知，其计算量减少为 $O(N^3)$。

● **基底图像** (Basis Image)

令 $\boldsymbol{a}_k^*$ 表示矩阵 $\boldsymbol{A}^{*\mathrm{T}}$ 的第 k 行，并定义矩阵

$$\boldsymbol{A}_{k,l}^* = \boldsymbol{a}_k^*\boldsymbol{a}_l^{\mathrm{T}} \tag{4.1-11}$$

且两个 $N\times N$ 矩阵 $\boldsymbol{F}$ 和 $\boldsymbol{G}$ 的内积为

$$\langle \boldsymbol{F}, \boldsymbol{G} \rangle = \sum_{m=0}^{N-1}\sum_{n=0}^{N-1} f(m,n)g^*(m,n) \tag{4.1-12}$$

则由 (4.1-1) 和 (4.1-2) 式可将图像表示成级数

$$f = \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} F(k,l) A_{k,l}^{*} \tag{4.1-13}$$

$$F(k,l) = \left\langle f, A_{k,l}^{*} \right\rangle \tag{4.1-14}$$

从 (4.1-13) 式可知任何图像f都是N^2个矩阵$A_{k,l}^{*}, k, l = 0,1,\cdots,N-1$的线性组合，因此$A_{k,l}^{*}$称为基底图像。变换系数$F(k,l)$是图像$f$与基底图像$A_{k,l}^{*}$的内积，所以又称为该基底图像的投影。

以下将介绍几种较常被采用的正交图像变换，包括傅立叶、离散余弦、离散正弦、Walsh-Hadamard、Haar、Slant、KL、Hartly、SVD 与小波变换。

4.2 傅立叶变换

4.2-1 离散傅立叶变换

离散信号$f(n)$的一维离散傅立叶变换 (discrete Fourier transform, DFT) 如下所示

$$F(k) = \sum_{k=0}^{N-1} f(n) e^{-j\left(\frac{2\pi}{N}\right)nk}, \quad k = 0,1,\cdots,N-1 \tag{4.2-1}$$

也可写成

$$F(k) = \sum_{n=0}^{N-1} f(n) W_N^{nk} \tag{4.2-2}$$

其中$W_N = e^{-j\frac{2\pi}{N}}$。$W_N^{nk}$通常被称为变换核 (kernel)。而$F(k)$的反离散傅立叶变换 (IDFT)则如下所示：

$$f(n) = \frac{1}{N} \sum_{k=0}^{N-1} F(k) e^{j\left(\frac{2\pi}{N}\right)nk}, \quad n = 0,1,\cdots,N-1 \tag{4.2-3}$$

因此，形成下面的变换对

$$f(n) \Leftrightarrow F(k) \tag{4.2-4}$$

一维离散傅立叶变换有以下的性质：

1. 线性性 (Linearity)

$$f_3(n) = af_1(n) + bf_2(n) \Leftrightarrow F_3(k) = aF_1(k) + F_2(k) \tag{4.2-5}$$

其中a和b为常数。

2. 对称性 (Symmetry)

若 $f(n)$ 为实数，则

$$F(N-k)=F^{*}(k) \tag{4.2-6}$$

3. 与卷积的关系

对于两个长度分别为 N_1 及 N_2 的离散信号 $f_1(n)$ 与 $f_2(n)$，执行第二章中所定义的卷积 $f_3(n)=f_1(n)*f_2(n)$ 后，$f_3(n)$ 的长度可达 $N=N_1+N_2-1$。现在将 $f_1(n)$ 与 $f_2(n)$ 分别添补 N_2-1 及 N_1-1 个零后，将其视为长度为 N 的两个新序列 $f_{1,N}(n)$ 与 $f_{2,N}(n)$，且其 N 点的 DFT 分别为 $F_1(k)$ 及 $F_2(k)$。则

$$f_3(n)=\mathrm{IDFT}[F_1(k)*F_2(k)] \tag{4.2-7}$$

换言之，这是通过 DFT 执行卷积的间接方法。配合下一节所讨论的快速傅立叶变换，当所考虑信号的长度大到一个程度以后，以 (4.2-7) 式执行卷积会比用定义直接计算快，而且信号长度愈长，所节省的时间愈多。

二维离散傅立叶变换具备与一维情况完全相同的性质。

4.2-2　快速傅立叶变换

快速傅立叶变换 (fast Fourier transform, FFT) 的算法，可分为时域抽取法 (decomposition- 或 decimation-in-time, DIT) 和频域抽取法 (decomposition- 或 decimation-in-frequency, DIF) 两种。

● **DIT 的算法**

兹以 $N=16$ 的离散函数 $f(n)$ 为例说明之。因为 $N=16$，所以

$$F(k)=\sum_{n=0}^{15} f(n)W_{16}^{nk},\quad k=0,1,\cdots,15 \tag{4.2-8}$$

依偶数位置和奇数位置，将序列 $f(n)$ 分成两个序列。可得

$$F(k)=\sum_{\substack{n=0\\n=\text{even}}}^{15} f(n)W_{16}^{nk}+\sum_{\substack{n=0\\n=\text{odd}}}^{15} f(n)W_{16}^{nk} \tag{4.2-9}$$

因为

$$W_{16}^{k(2n)}=\mathrm{e}^{-\mathrm{j}\frac{2\pi}{16}\cdot k(2n)}=\mathrm{e}^{-\mathrm{j}\frac{2\pi}{8}\cdot nk} \tag{4.2-10}$$

所以

$$F(k)=\sum_{n=0}^{7} f(2n)W_{8}^{nk}+W_{16}^{k}\sum_{n=0}^{7} f(2n+1)W_{8}^{nk} \tag{4.2-11}$$

若令

$$f_{10}(n) = f(2n) \tag{4.2-12}$$

$$f_{11}(n) = f(2n+1) \tag{4.2-13}$$

则

$$F(k) = \sum_{n=0}^{7} f_{10}(n) W_8^{nk} + W_{16}^{k} \sum_{n=0}^{7} f_{11}(n) W_8^{nk} \tag{4.2-14}$$

令

$$F_{10}(r) = \sum_{n=0}^{7} f_{10}(n) W_8^{nr} \tag{4.2-15}$$

$$F_{11}(r) = \sum_{n=0}^{7} f_{11}(n) W_8^{nr} \tag{4.2-16}$$

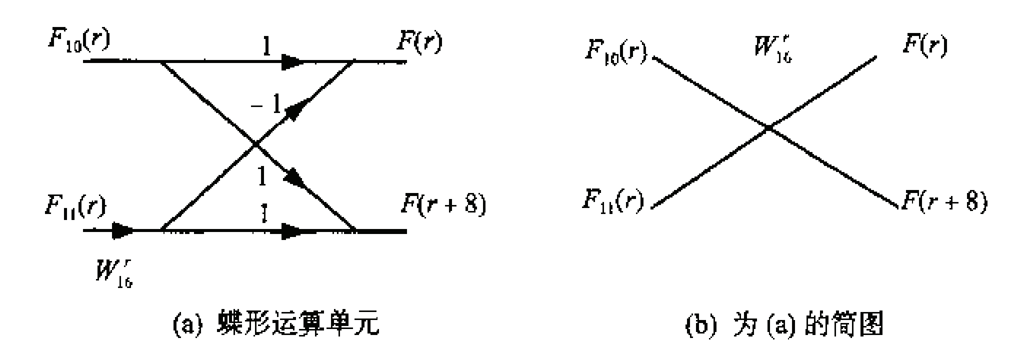
$r = 0,1,\cdots,7$. 即分别为$f_{10}(n)$ 及$f_{11}(n)$ 的 8 点 DFT。重写 (4.2-14) 式，可得

$$F(r) = F_{10}(r) + W_{16}^{r} F_{11}(r) \tag{4.2-17}$$

另外、由 $W_8^{n(r+8)} = W_8^{nr}$ 也可轻易证得

$$F(r+8) = F_{10}(r) - W_{16}^{r} F_{11}(r) \tag{4.2-18}$$

$r = 0,1,\cdots,7$。可见 (4.2-17) 及 (4.2-18) 式形成一个运算单元，即所谓的**蝶形** (butterfly) 运算单元。在图 4.2-1 中，用图形来代表一个蝶形运算单元，其中 W_{16}^{r} 这一项通常称为**权重** (weight) 或**旋转因子** (twiddle factor)。(4.2-17) 及 (4.2-18) 式代表整个 FFT 的最后一级，而前三级可依此类推。亦即一个 16 点的 DFT 可由两个 8 点 DFT 的碟形组合而成；各 8 点的 DFT 可由两个 4 点 DFT 的碟形组合而成；各 4 点的 DFT 可由两个 2 点 DFT 的碟形组合而成；整个 DIT 算法的流程图如图 4.2-2 所示。



(a) 蝶形运算单元 (b) 为 (a) 的简图

图 4.2-1 FFT 的 DIT 运算单元

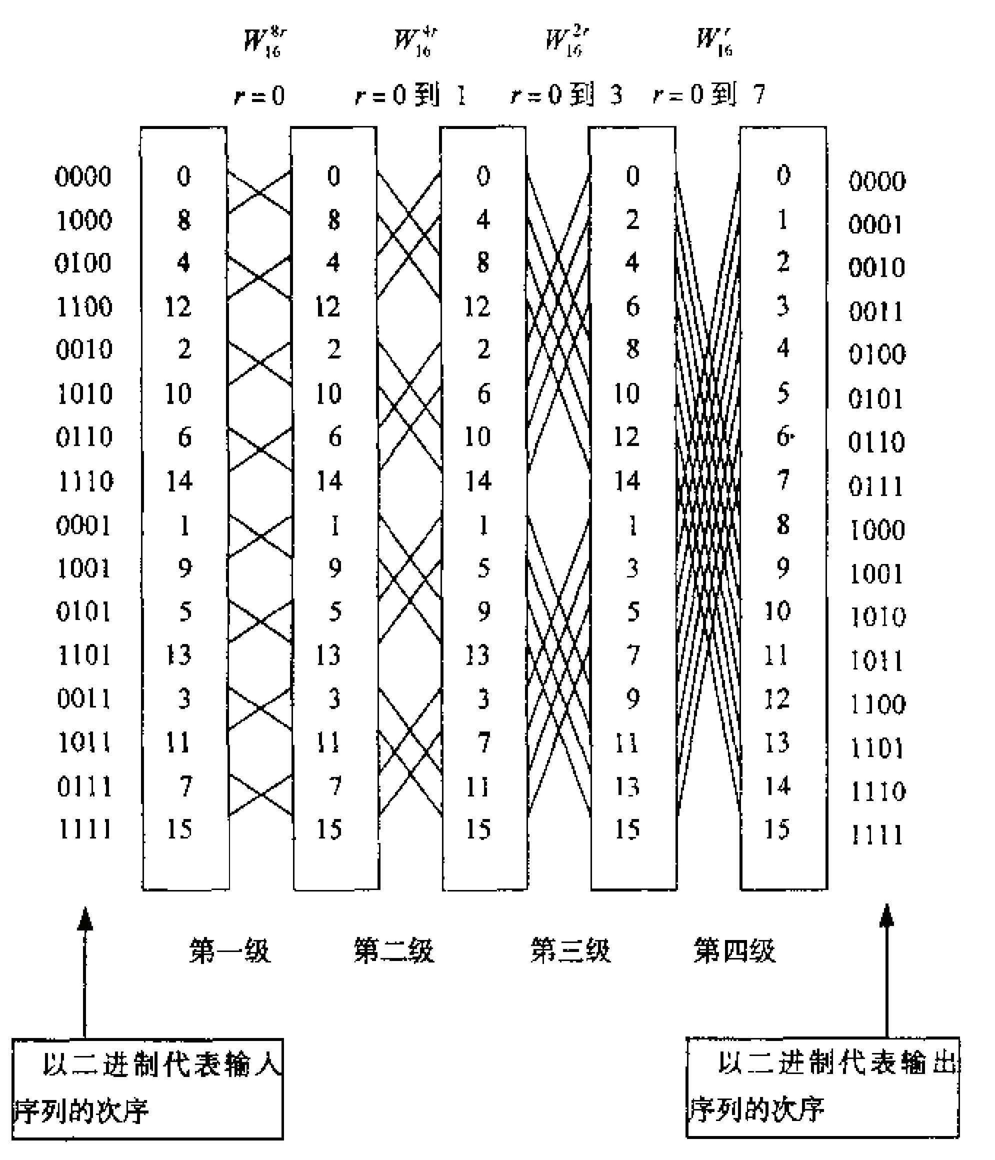


图 4.2-2　整个 DIT 算法的流程图

图中输出入序列的序号间恰有位倒置的关系。所谓位倒置就是将一序列的序号的二进制码反序后作为新的序号。下面举例说明。

序号	0	1	2	3	4	5	6	7
二进制	000	001	010	011	100	101	110	111
反序	000	100	010	110	001	101	011	111
新序	0	4	2	6	1	5	3	7

● **DIF 的算法**

前面的 DIT 是将输入序列分奇偶位置打散，但在 DIF 的算法中则改将输出序列打散，而其他过程则相同。

我们再度以 $N = 16$ 的离散函数 $f(n)$ 为例说明之。因为 $N = 16$，所以

$$F(k) = \sum_{n=0}^{15} f(n) W_{16}^{nk}, \quad k = 0,1,\cdots,15 \tag{4.2-19}$$

将上式依 $f(n)$ 拆成两部分，即

$$F(k)=\sum_{n=0}^{7}f(n)W_{16}^{nk}+\sum_{n=8}^{15}f(n)W_{16}^{nk} \tag{4.2-20}$$

其中第二个和可写成

$$\begin{aligned}\sum_{n=8}^{15}f(n)W_{16}^{nk}&=\sum_{n=0}^{7}f(n+8)W_{16}^{(n+8)k}\\&=(-1)^{k}\sum_{n=0}^{7}f(n+8)W_{16}^{nk}\end{aligned} \tag{4.2-21}$$

将此结果代回 (4.2-20) 式可得

$$F(k)=\sum_{n=0}^{7}[f(n)+(-1)^{k}f(n+8)]W_{16}^{nk} \tag{4.2-22}$$

由此得

$$\begin{aligned}F(2k)&=\sum_{n=0}^{7}[f(n)+f(n+8)]W_{16}^{2kn}, \qquad k=0,1,\cdots,7\\F(2k+1)&=\sum_{n=0}^{7}[f(n)-f(n+8)]W_{16}^{(2k+1)n}, \qquad k=0,1,\cdots,7\end{aligned} \tag{4.2-23}$$

利用

$$W_{16}^{2kn}=\mathrm{e}^{-\mathrm{j}(\frac{2\pi}{16})2nk}=\mathrm{e}^{-\mathrm{j}(\frac{2\pi}{8})nk}=W_{8}^{kn} \tag{4.2-24}$$

以及

$$W_{16}^{(2k+1)n}=\mathrm{e}^{-\mathrm{j}\left(\frac{2\pi}{16}\right)(2k+1)n}=\mathrm{e}^{-\mathrm{j}\frac{2\pi}{16}n}\cdot\mathrm{e}^{-\mathrm{j}\left(\frac{2\pi}{16}\right)2kn}=W_{16}^{n}W_{8}^{kn} \tag{4.2-25}$$

可将 (4.2-23) 式改写为

$$\left.\begin{aligned}F(2k)&=\sum_{n=0}^{7}[f(n)+f(n+8)]W_{8}^{kn}, \qquad k=0,1,\cdots,7\\F(2k+1)&=\sum_{n=0}^{7}[f(n)-f(n+8)]W_{16}^{n}W_{8}^{kn}, \qquad k=0,1,\cdots,7\end{aligned}\right\} \tag{4.2-26}$$

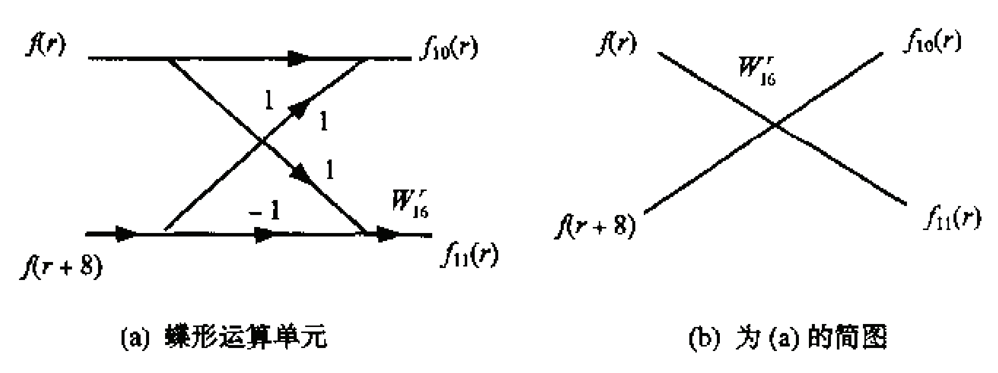
若令

$$f_{10}(r)=f(r)+f(r+8)\text{ 且 }f_{11}(r)=[f(r)-f(r+8)]W_{16}^{r} \tag{4.2-27}$$

则

$$F(2k)=\sum_{r=0}^{7} f_{10}(r)W_8^{kr}, \quad k=0,1,\cdots 7$$
$$F(2k+1)=\sum_{r=0}^{7} f_{11}(r)W_8^{kr}, \quad k=0,1,\cdots 7 \tag{4.2-28}$$

亦即 $F(2k)$是 $f_{10}(r)$ 的 8 点 DFT，$F(2k+1)$ 则是$f_{11}(r)$ 的 8 点 DFT。8 点 DFT 又可拆成 2 个 4 点 DFT，依此类推。将 (4.2-27) 式以蝶形运算单元的形式呈现如图 4.2-3 所示的结果。整个 DIF 算法的流程图则如图 4.2-4 所示。



(a) 蝶形运算单元　　　　(b) 为 (a) 的简图

图 4.2-3　FFT 的 DIF 运算单元

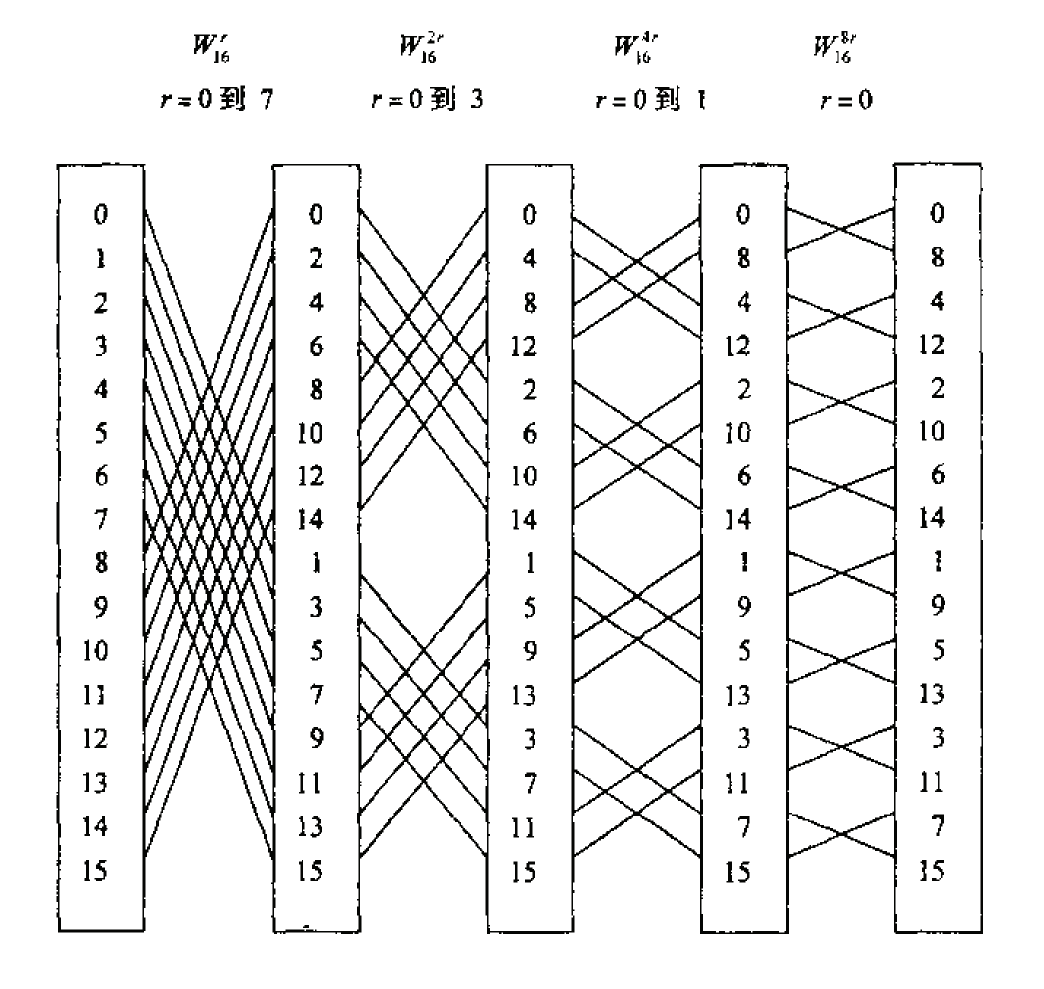


图 4.2-4　16 点 FFT 的 DIF 流程图

由 FFT 算法的流程图可知计算复杂度：FFT 算法仅需 $(N/2)\log_2 N$ 的复数乘积，远小于 DFT 的 N^2 个复数乘积。当 N 够大时，执行效率明显提升。

4.3 离散余弦变换

离散余弦变换 (discrete cosine transform, DCT) 是离散傅立叶变换的一员；与 DFT 相同的是，离散余弦变换提供了有关信号在频域的信息；与 DFT 不同的是，一个实数信号的离散余弦变换具有实数值。因 DCT 变换为线性，故可表示成如下的矩阵向量形式

$$\boldsymbol{F}=\boldsymbol{Cf} \tag{4.3-1}$$

此处 $\boldsymbol{f}$ 是描述信号的 N 维向量，$\boldsymbol{F}$ 是描述变换结果的 N 维向量，$\boldsymbol{C}$ 是描述变换的 $N\times N$ 非奇异实数矩阵。若 $\boldsymbol{f}$ 为 $N\times N$ 的图像矩阵，则其 DCT 的矩阵形式为

$$\boldsymbol{F}=\boldsymbol{CfC}^{\mathrm{T}} \tag{4.3-2}$$

4.3-1 第一类 DCT

设 $f(n)$ 为长度 N 的离散时间实数信号，将此信号连同其对称复制信号定义出如下的新信号 $f_1(n)$：

$$f_1(n)=\begin{cases}2^{\frac{1}{2}}f(n), & n=0,N-1\\ f(n), & 1\leqslant n\leqslant N-2\\ f(2N-n-2), & N\leqslant n\leqslant 2N-3\end{cases} \tag{4.3-3}$$

$f_1(n)$ 的 DFT 为

$$\begin{aligned}F_1(k)&=\sum_{n=0}^{2N-3}f_1(n)W_{2N-2}^{nk}=2^{\frac{1}{2}}f(0)+2^{\frac{1}{2}}f(N-1)(-1)^k\\&\quad+\sum_{n=1}^{N-2}f(n)W_{2N-2}^{nk}+\sum_{n=N}^{2N-3}f(2N-n-2)W_{2N-2}^{nk}\\&=2^{\frac{1}{2}}f(0)+2^{\frac{1}{2}}f(N-1)(-1)^k+\sum_{n=1}^{N-2}f(n)\left(W_{2N-2}^{nk}+W_{2N-2}^{-nk}\right)\\&=2^{\frac{1}{2}}f(0)+2^{\frac{1}{2}}f(N-1)(-1)^k+2\sum_{n=1}^{N-2}f(n)\cos\left(\frac{\pi nk}{N-1}\right)\\&=2\sum_{n=0}^{N-1}a(n)f(n)\cos\left(\frac{\pi nk}{N-1}\right),\qquad 0\leqslant k\leqslant 2N-3\end{aligned} \tag{4.3-4}$$

此处

$$a(n)=\begin{cases}2^{-\frac{1}{2}}, & n=0,N-1\\ 1, & \text{其他情况}\end{cases} \tag{4.3-5}$$

序列 $F_1(k)$ 为实数且满足 $F_1(k)=F_1(2N-n-k)$，所以

$$\begin{aligned}f_1(n)&=\frac{1}{2N-2}\sum_{k=0}^{2N-3}F_1(k)W_{2N-2}^{-nk}\\&=\frac{1}{2N-2}F_1(0)+\frac{1}{2N-2}F_1(N-1)(-1)^n+\frac{1}{N-1}\sum_{k=1}^{N-2}F_1(k)\cos\left(\frac{\pi nk}{N-1}\right)\end{aligned} \tag{4.3-6}$$

根据上述结果，第一类 DCT 定义如下

$$F(k)=\frac{1}{\sqrt{2(N-1)}}a(k)F_1(k)=\sqrt{\frac{2}{N-1}}\sum_{n=0}^{N-1}a(k)a(n)f(n)\cos\left(\frac{\pi nk}{N-1}\right) \tag{4.3-7}$$

从 (4.3-7) 式可得 $f(n)=\dfrac{1}{\sqrt{2(N-1)}}a(n)f_1(n)$，连同 (4.3-6) 式可得反变换如下

$$f(n)=\sqrt{\frac{2}{N-1}}\sum_{k=0}^{N-1}a(k)a(n)F(k)\cos\left(\frac{\pi nk}{N-1}\right) \tag{4.3-8}$$

我们也可以将第一类 DCT 的正反变换用矩阵向量运算表示

$$\boldsymbol{F}=\boldsymbol{C}_{\mathrm{I}}\boldsymbol{f},\quad \boldsymbol{f}=\boldsymbol{C}_{\mathrm{I}}\boldsymbol{F} \tag{4.3-9}$$

此处，

$$[\boldsymbol{C}_{\mathrm{I}}]_{k,n}=\sqrt{\frac{2}{N-1}}a(k)a(n)\cos\left(\frac{\pi nk}{N-1}\right),\quad 0\leqslant k,n\leqslant N-1 \tag{4.3-10}$$

$\boldsymbol{C}_{\mathrm{I}}$ 称为 DCT-I 矩阵、从其定义中可清楚看出这个矩阵是对称的。另外从 (4.3-9) 式可知 $\boldsymbol{C}_{\mathrm{I}}$ 亦是本身的逆矩阵，也就是

$$(\boldsymbol{C}_{\mathrm{I}})^{-1}=\boldsymbol{C}_{\mathrm{I}} \tag{4.3-11}$$

因此 DCT-I 矩阵为一对称的么正矩阵。

4.3-2　第二类 DCT

第二类 DCT 是以所有 N 点都被复制的 $f(n)$ 信号的最小对称着手。我们定义

$$f_2(n)=\begin{cases}f(n), & 0\leqslant n\leqslant N-1\\ f(2N-1-n), & N\leqslant n\leqslant 2N-1\end{cases} \tag{4.3-12}$$

$f_2(n)$ 的 DFT 为

$$\begin{aligned} F_2(k) &= \sum_{n=0}^{N-1} f(n) W_{2N}^{nk} + \sum_{n=N}^{2N-1} f(2N-1-n) W_{2N}^{nk} \\ &= \sum_{n=0}^{N-1} f(n)\left(W_{2N}^{nk} + W_{2N}^{-(n+1)k}\right) \end{aligned} \tag{4.3-13}$$

这个序列不为实数，对 k 值亦不为偶对称；不过，如果再乘上 $W_{2N}^{0.5k}$ ，也就是

$$\begin{aligned} U(k) = W_{2N}^{0.5k} F_2(k) &= \sum_{n=0}^{N-1} f(n)\left(W_{2N}^{(n+0.5)k} + W_{2N}^{-(n+0.5)k}\right) \\ &= 2\sum_{n=0}^{N-1} f(n)\cos\left[\frac{\pi(n+0.5)k}{N}\right], \quad 0 \leqslant k \leqslant 2N-1 \end{aligned} \tag{4.3-14}$$

则所得的新序列 $U(k)$ 有如下性质：

1. 序列 $U(k)$ 满足奇对称关系式

$$U(2N-k) = -U(k) \tag{4.3-15}$$

也就是

$$\cos\left[\frac{\pi(n+0.5)(2N-k)}{N}\right] = -\cos\left[\frac{\pi(n+0.5)k}{N}\right] \tag{4.3-16}$$

2. $U(N)=0$ ，也就是

$$\cos\left[\frac{\pi(n+0.5)N}{N}\right] = \cos\left[\pi(n+0.5)\right] = 0 \tag{4.3-17}$$

原始序列可用 $U(k)$ 表示成

$$\begin{aligned} f(n) &= \frac{1}{2N}\sum_{k=0}^{2N-1} F_2(k) W_{2N}^{-nk} = \frac{1}{2N}\sum_{k=0}^{2N-1} U(k) W_{2N}^{-0.5k} W_{2N}^{-nk} \\ &= \frac{1}{2N}\sum_{k=0}^{N-1} U(k) W_{2N}^{-(n+0.5)k} - \frac{1}{2N}\sum_{k=N+1}^{2N-1} U(2N-k) W_{2N}^{-(n+0.5)k} \\ &= \frac{1}{2N}U(0) + \frac{1}{2N}\sum_{k=1}^{N-1} U(k)\left(W_{2N}^{-(n+0.5)k} + W_{2N}^{(n+0.5)k}\right) \\ &= \frac{1}{2N}U(0) + \frac{1}{N}\sum_{k=1}^{N-1} U(k)\cos\left[\frac{\pi(n+0.5)k}{N}\right] \end{aligned} \tag{4.3-18}$$

变换 DCT-II 定义为

$$F(k) = \frac{1}{\sqrt{2N}} b(k) U(k) \tag{4.3-19}$$

此处

$$b(k)=\begin{cases}2^{\frac{-1}{2}}, & k=0\\ 1, & k\neq 0\end{cases} \tag{4.3-20}$$

将 (4.3-14) 与 (4.3-20) 式代人 (4.3-19) 式可得

$$F(k)=\sqrt{\frac{2}{N}}\sum_{n=0}^{N-1}b(k)f(n)\cos\left[\frac{\pi(n+0.5)k}{N}\right] \tag{4.3-21}$$

将 (4.3-14) 式及 (4.3-20) 式代人 (4.3-18) 式可得

$$f(n)=\sqrt{\frac{2}{N}}\sum_{k=0}^{N-1}b(k)F(k)\cos\left[\frac{\pi(n+0.5)k}{N}\right] \tag{4.3-22}$$

其中 (4.3-21) 式及 (4.3-22) 式为 DCT-II 变换对，其矩阵向量表示式为

$$\boldsymbol{F}=\boldsymbol{C}_{\mathrm{II}}\boldsymbol{f},\quad \boldsymbol{f}=(\boldsymbol{C}_{\mathrm{II}})^{\mathrm{T}}\boldsymbol{F} \tag{4.3-23}$$

此处

$$[\boldsymbol{C}_{\mathrm{II}}]_{k,n}=\sqrt{\frac{2}{N}}b(k)\cos\left[\frac{\pi(n+0.5)k}{N}\right],\quad 0\leqslant k,n\leqslant N-1 \tag{4.3-24}$$

根据 (4.3-23) 式可知 $\boldsymbol{C}_{\mathrm{II}}$ 为一实数规范正交矩阵，也就是它满足

$$(\boldsymbol{C}_{\mathrm{II}})^{-1}=(\boldsymbol{C}_{\mathrm{II}})^{\mathrm{T}} \tag{4.3-25}$$

4.3-3　第三类 DCT

第三类 DCT 为第二类 DCT 的逆矩阵或转置矩阵，定义为

$$F(k)=\sqrt{\frac{2}{N}}\sum_{n=0}^{N-1}b(n)f(n)\cos\left[\frac{\pi(k+0.5)n}{N}\right] \tag{4.3-26}$$

$$f(n)=\sqrt{\frac{2}{N}}\sum_{k=0}^{N-1}b(n)F(k)\cos\left[\frac{\pi(k+0.5)n}{N}\right] \tag{4.3-27}$$

因此 DCT-III 与 DCT-II 矩阵有如下关系

$$\boldsymbol{C}_{\mathrm{III}}=(\boldsymbol{C}_{\mathrm{II}})^{\mathrm{T}} \tag{4.3-28}$$

4.3-4　第四类 DCT

DCT-II 与 DCT-III 在 k 与 n 都不为对称，但 DCT-IV 与 DCT-I 一样都为对称，它的结构

式与 DCT-II 相似，除了将 $f(n)$ 对称展开外，我们将它调整如下

$$f_4(n)=\begin{cases} W_{2N}^{0.5n} f(n), & 0 \leq n \leq N-1 \\ -W_{2N}^{0.5n} f(2N-1-n), & N \leq n \leq 2N-1 \end{cases} \tag{4.3-29}$$

DCT-IV 定义为

$$F(k)=\sqrt{\frac{2}{N}}\sum_{n=0}^{N-1} f(n)\cos\left[\frac{\pi(n+0.5)(k+0.5)}{N}\right] \tag{4.3-30}$$

其反变换式为

$$f(n)=\sqrt{\frac{2}{N}}\sum_{k=0}^{N-1} F(k)\cos\left[\frac{\pi(n+0.5)(k+0.5)}{N}\right] \tag{4.3-31}$$

(4.3-30) 式及 (4.3-31) 式的推导过程当做习题。其矩阵向量表示式为

$$\boldsymbol{F}=\boldsymbol{C}_{\mathrm{IV}} \boldsymbol{f}, \quad \boldsymbol{f}=\boldsymbol{C}_{\mathrm{IV}} \boldsymbol{F} \tag{4.3-32}$$

此处

$$[\boldsymbol{C}_{\mathrm{IV}}]_{k,n}=\sqrt{\frac{2}{N}}\cos\left[\frac{\pi(n+0.5)(k+0.5)}{N}\right], \quad 0 \leq k,n \leq N-1 \tag{4.3-33}$$

从 (4.3-32) 式可知：$\boldsymbol{C}_{\mathrm{IV}}$ 为一对称的规范正交矩阵，亦即它满足

$$(\boldsymbol{C}_{\mathrm{IV}})^{-1}=\boldsymbol{C}_{\mathrm{IV}} \tag{4.3-34}$$

4.4 离散正弦变换

离散正弦变换 (discrete sine transform, DST) 与 DCT 相似，它被视为**傅立叶正弦级数** (Fourier sine series) 的离散时间版本，共有四种类型的 DST。DST 的推演过程与 DCT 的推演过程类似， 因此我们只定义 DST，略过其推导的细节。在所有情况下，$0 \leq n,k \leq N-1$。

1. DST-I

$$[\boldsymbol{S}_{\mathrm{I}}]_{k,n}=\sqrt{\frac{2}{N+1}}\sin\left[\frac{\pi(k+1)(n+1)}{N+1}\right], \quad (\boldsymbol{S}_{\mathrm{I}})^{-1}=\boldsymbol{S}_{\mathrm{I}} \tag{4.4-1}$$

2. DST-II

$$[\boldsymbol{S}_{\mathrm{II}}]_{k,n}=\sqrt{\frac{2}{N}}c(k)\sin\left[\frac{\pi(k+1)(n+0.5)}{N+1}\right], \quad (\boldsymbol{S}_{\mathrm{II}})^{-1}=(\boldsymbol{S}_{\mathrm{II}})^{T} \tag{4.4-2}$$

3. DST-III

$$[\boldsymbol{S}_{\mathrm{III}}]_{k,n}=\sqrt{\frac{2}{N}}c(k)\sin\left[\frac{\pi(k+0.5)(n+1)}{N+1}\right],\quad (\boldsymbol{S}_{\mathrm{III}})^{-1}=(\boldsymbol{S}_{\mathrm{III}})^{T} \tag{4.4-3}$$

4. DST-IV

$$[\boldsymbol{S}_{\mathrm{IV}}]_{k,n}=\sqrt{\frac{2}{N}}\sin\left[\frac{\pi(k+0.5)(n+0.5)}{N+1}\right],\quad (\boldsymbol{S}_{\mathrm{IV}})^{-1}=\boldsymbol{S}_{\mathrm{IV}} \tag{4.4-4}$$

在上述定义中

$$c(n)=\begin{cases}2^{-\frac{1}{2}}, & n=N-1\\ 0, & n\neq N-1\end{cases} \tag{4.4-5}$$

4.5　Walsh-Hadamard 变换

Walsh 函数是一组定义在 [0,1] 上的完备正交矩形函数。离散 Walsh 函数是在 [0,1] 区间上对连续 Walsh 函数等间隔取样的结果。在图像处理中，取样点数 N 常取 2 的正整数乘幂，此时 Hadamard 变换矩阵与 Walsh 变换矩阵只在行（或列）的次序上不同，因此 Walsh 变换与 Hadamard 变换常统称为“Walsh-Hadamard”变换。对任意的正整数 N，Walsh 变换都可形成，但对于不为 2 的正整数乘幂的 N 值，Hadamard 变换不一定可形成。

此处只考虑 N 为 2 的正整数乘幂的状况，此时 Hadamard 与 Walsh 的变换矩阵只有排列顺序不同。为了呈现此种差别，首先我们介绍列率 (sequency) 的观念，再分别探讨 Walsh 版以及 Hadamard 版的 Walsh-Hadamard 变换。

4.5-1　列　率

频率可定义为一个正弦函数在每单位时间内所经历的周期数，它正好是函数值越零 (zero crossing) 次数的一半。

频率的概念可以推广至称为列率的广义频率，列率可定义为函数在单位时间内其值越零点的平均次数的一半。列率可用来描述在一个区间内有不等间隔的越零点的非周期性函数。对于周期性的正弦函数，列率与频率的定义相当。

图 4.5-1 显示在一个区间[0,1)上的一个连续函数 $f(x)$。由于该区间上函数有四个越零点，所以列率等于 2。

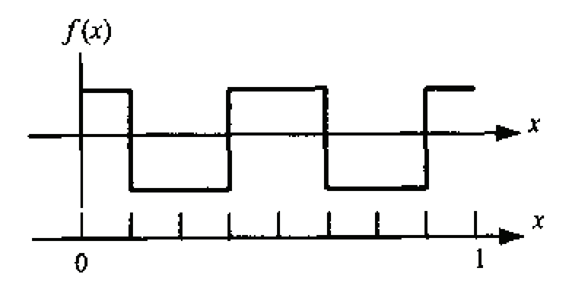


图 4.5-1　连续函数列率的例子

对于离散函数 $f(n)$，列率 S 可定义为

$$S=\begin{cases}\dfrac{r}{2}, & r\text{ 为偶数}\\ \dfrac{r+1}{2}, & r\text{ 为奇数}\end{cases} \tag{4.5-1}$$

其中 r 为在单位时间内 $f(n)$ 的符号变更次数。对图 4.5-1 中的 $f(x)$ 做等间隔取样，得到一个离散函数 $f(n)$，如图 4.5-2 中所示。从图中可看出其符号变更次数为 $r=4$，故其列率为 2。

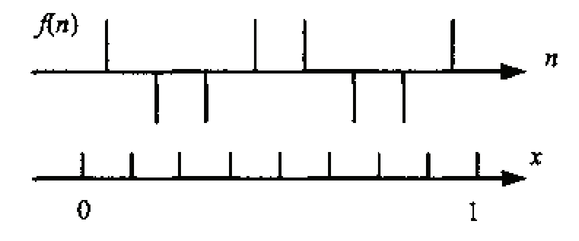


图 4.5-2 离散函数列率的例子

4.5-2 Walsh 版的 Walsh-Hadamard 变换

以 $N=8$ 为例，Walsh 版的 Walsh-Hadamard 变换 (记为 WWHT) 的变换矩阵 $\boldsymbol{H}_8$ 为

$$\boldsymbol{H}_8=\frac{1}{2\sqrt{2}}\begin{bmatrix}1&1&1&1&1&1&1&1\\1&1&1&1&-1&-1&-1&-1\\1&1&-1&-1&-1&-1&1&1\\1&1&-1&-1&1&1&-1&-1\\1&-1&-1&1&1&-1&-1&1\\1&-1&-1&1&-1&1&1&-1\\1&-1&1&-1&-1&1&-1&1\\1&-1&1&-1&1&-1&1&-1\end{bmatrix}\begin{matrix}\text{变号次数}\\0\\1\\2\\3\\4\\5\\6\\7\end{matrix} \tag{4.5-2}$$

其变换系数指标 k、变号次数 r 及列率 S 的关系如表 4.5-1 所示。

表 4.5-1 WWHT 的 $\boldsymbol{H}_8$ 的列率

变换系数指标 k	0	1	2	3	4	5	6	7
变号次数 r	0	1	2	3	4	5	6	7
列率 S	0	1	1	2	2	3	3	4

规一化后，可轻易验证出 $\boldsymbol{H}_N$ 为一对称的正交矩阵，因此

$$\boldsymbol{H}_N^{-1}=\boldsymbol{H}_N \tag{4.5-3}$$

由此可知 WWHT 的正反变换有相同的变换矩阵。其一维正反变换核 $h(k,n)$ 为

$$h(k,n)=\frac{1}{\sqrt{N}}(-1)^{p(k,n)},\quad k,n=0,1,\cdots,N-1 \tag{4.5-4}$$

$$p(k,n)=\sum_{i=0}^{l-1}t_i(k)b_i(n) \tag{4.5-5}$$

其中 $b_i(n)$ 为以二进制数代表整数 n 时，第 i 位的值，l 是使 $N=2^l$ 的值。另外，若以符号 $\oplus$ 表示“XOR”(2 余数加法) 运算，则 $t_i(k)$ 的定义为

$$\begin{cases}t_0(k)=b_{l-1}(k)\\ t_1(k)=b_{l-1}(k)\oplus b_{l-2}(k)\\ t_2(k)=b_{l-2}(k)\oplus b_{l-3}(k)\\ \qquad\vdots\\ t_{l-1}(k)=b_1(k)\oplus b_0(k)\end{cases} \tag{4.5-6}$$

以 $(n,k)=(1,1)$ 与 $(3,6)$ 为例，(4.5-4) 到 (4.5-6) 式的计算结果如表 4.5-2 中所示。

表 4.5-2　(4.5-4) 到 (4.5-6) 式的计算实例

n	$b_2(n)$	$b_1(n)$	$b_0(n)$	k	$b_2(k)$	$b_1(k)$	$b_0(k)$	$t_2(k)$	$t_1(k)$	$t_0(k)$	$p(k,n)$	$h(k,n)$
1	0	0	1	1	0	0	1	1	0	0	0	$\frac{1}{2\sqrt{2}}$
3	0	1	1	6	1	1	0	1	0	1	1	$\frac{1}{2\sqrt{2}}$

一维 WWHT 的正反变换为

$$F(k)=\sum_{n=0}^{N-1}f(n)h(k,n)=\frac{1}{\sqrt{N}}\sum_{n=0}^{N-1}f(n)(-1)^{\sum_{i=0}^{l-1}t_i(k)b_i(n)} \tag{4.5-7}$$

$$f(n)=\sum_{k=0}^{N-1}F(k)h(k,n)=\frac{1}{\sqrt{N}}\sum_{k=0}^{N-1}F(k)(-1)^{\sum_{i=0}^{l-1}t_i(k)b_i(n)} \tag{4.5-8}$$

表示成矩阵形式分别可得

$$\boldsymbol{F}=\boldsymbol{H}_N\boldsymbol{f} \tag{4.5-9}$$

$$\boldsymbol{f}=\boldsymbol{H}_N\boldsymbol{F} \tag{4.5-10}$$

对一 $M\times N$ 图像，其二维 WWHT 的正反变换核为

$$h(j,k;m,n)=\frac{1}{\sqrt{MN}}(-1)^{p(j,m)+p(k,n)} \tag{4.5-11}$$

$$p(j,m)+p(k,n)=\sum_{i=0}^{J-1}\left[t_i(j)b_i(m)+t_i(k)b_i(n)\right] \tag{4.5-12}$$

由于 $h(j,k;m,n)$ 是可分离的，即

$$h(j,k;m,n)=h_1(j,m)h_2(k,n) \tag{4.5-13}$$

故二维 WWHT 变换的矩阵形式为

$$\boldsymbol{F}=\boldsymbol{H}_M\boldsymbol{f}\boldsymbol{H}_N \tag{4.5-14}$$

$$\boldsymbol{f}=\boldsymbol{H}_M\boldsymbol{F}\boldsymbol{H}_N \tag{4.5-15}$$

相对应的变换式为

$$F(j,k)=\frac{1}{\sqrt{MN}}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}f(m,n)(-1)^{\sum_{i=0}^{J-1}[t_i(j)b_i(m)+t_i(k)b_i(n)]} \tag{4.5-16}$$

$$f(m,n)=\frac{1}{\sqrt{MN}}\sum_{j=0}^{M-1}\sum_{k=0}^{N-1}F(j,k)(-1)^{\sum_{i=0}^{J-1}[t_i(j)b_i(m)+t_i(k)b_i(n)]} \tag{4.5-17}$$

4.5-3 Hadamard 版的 Walsh-Hadamard 变换

以 $N=4$ 和 $N=8$ 为例，Hadamard 版的 Walsh-Hadamard 变换（记为 HWHT）的两个变换矩阵为

$$\boldsymbol{H}_4=\frac{1}{2}\begin{bmatrix}1&1&1&1\\1&-1&1&-1\\1&1&-1&-1\\1&-1&-1&1\end{bmatrix}\quad\begin{matrix}\text{变号次数}\\0\\3\\1\\2\end{matrix} \tag{4.5-18}$$

$$\boldsymbol{H}_8=\frac{1}{2\sqrt{2}}\begin{bmatrix}1&1&1&1&1&1&1&1\\1&-1&1&-1&1&-1&1&-1\\1&1&-1&-1&1&1&-1&-1\\1&-1&-1&1&1&-1&-1&1\\1&1&1&1&-1&-1&-1&-1\\1&-1&1&-1&-1&1&-1&1\\1&1&-1&-1&-1&-1&1&1\\1&-1&-1&1&-1&1&1&-1\end{bmatrix}\quad\begin{matrix}\text{变号次数}\\0\\7\\3\\4\\1\\6\\2\\5\end{matrix} \tag{4.5-19}$$

其中 $\boldsymbol{H}_8$ 的变换系数指标 k，变号次数 r 及列率 S 的关系如表 4.5-3 所示。

表 4.5-3　$\boldsymbol{H}_8$ 的 列 率

变换系数 k	0	1	2	3	4	5	6	7
变号次数 r	0	7	3	4	1	6	2	5
列率 S	0	4	2	2	1	3	1	3

由表 4.5-3 可知，其列率的顺序并不像 WWHT 一样是递增的，但是这样的好处是其变换矩阵可由迭代关系得到，它是由 +1 和 −1 的元素构成的 $N=2^l$ 阶方阵。在 $N=2^l$ 时，形成该矩阵的迭代关系为

$$\boldsymbol{H}_2=\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} \tag{4.5-20}$$

$$\boldsymbol{H}_{2^l}=\frac{1}{\sqrt{2}}\begin{bmatrix}\boldsymbol{H}_{2^{l-1}} & \boldsymbol{H}_{2^{l-1}}\\ \boldsymbol{H}_{2^{l-1}} & -\boldsymbol{H}_{2^{l-1}}\end{bmatrix}=\boldsymbol{H}_2\otimes\boldsymbol{H}_{2^{l-1}} \tag{4.5-21}$$

读者可以此式验证 (4.5-18) 及 (4.5-19) 两式中的 $\boldsymbol{H}_4$ 与 $\boldsymbol{H}_8$。

经过规化后，矩阵 $\boldsymbol{H}_N$ 有下列性质：

1. $\boldsymbol{H}_N$ 是一对称矩阵，即

$$\boldsymbol{H}_N^{\mathrm{T}}=\boldsymbol{H}_N \tag{4.5-22}$$

2. $\boldsymbol{H}_N$ 是一正交矩阵，即

$$\boldsymbol{H}_N^{\mathrm{T}}\boldsymbol{H}_N=\boldsymbol{H}_N\boldsymbol{H}_N^{\mathrm{T}}=\boldsymbol{I}_N \tag{4.5-23}$$

由上两式可得 $\boldsymbol{H}_N^{-1}=\boldsymbol{H}_N$，这说明 HWHT 的正变换矩阵与反变换矩阵完全相同，即它们的正反变换核相同，因此其正、反变换的求和表示式也一致。

一维 HWHT 的变换核 $h(k,n)$ 为

$$h(k,n)=\frac{1}{\sqrt{N}}(-1)^{p(k,n)},\quad k,n=0,1,\cdots,N-1 \tag{4.5-24}$$

$$p(k,n)=\sum_{i=0}^{l-1}b_i(k)b_i(n) \tag{4.5-25}$$

其中 $b_i(k)$ 为以二进制数代表整数 k 时，第 i 位的值，见表 4.5-4。

表 4.5-4　二 进 制 编 码

k	$b_3(k)$	$b_2(k)$	$b_1(k)$	$b_0(k)$
14	1	1	1	0
10	1	0	1	0
7	0	1	1	1

在上述定义之下，一维 HWHT 的正、反变换的求和式分别为

$$F(k)=\sum_{n=0}^{N-1} f(n)h(k,n)=\frac{1}{\sqrt{N}}\sum_{n=0}^{N-1} f(n)(-1)^{\sum_{i=0}^{l-1} b_i(k)b_i(n)} \tag{4.5-26}$$

$$f(n)=\sum_{k=0}^{N-1} F(k)h(k,n)=\frac{1}{\sqrt{N}}\sum_{k=0}^{N-1} F(k)(-1)^{\sum_{i=0}^{l-1} b_i(k)b_i(n)} \tag{4.5-27}$$

以矩阵形式写出的 HWHT 正、反变换式是

$$\boldsymbol{F}=\boldsymbol{H}_N\boldsymbol{f} \tag{4.5-28}$$

$$\boldsymbol{f}=\boldsymbol{H}_N\boldsymbol{F} \tag{4.5-29}$$

对一$M\times N$的图像，其二维 HWHT 的正、反变换核为

$$h(j,k;m,n)=\frac{1}{\sqrt{MN}}(-1)^{p(j,m)+p(k,n)} \tag{4.5-30}$$

$$p(j,m)+p(k,n)=\sum_{i=0}^{l-1}\left[b_i(j)b_i(m)+b_i(k)b_i(n)\right] \tag{4.5-31}$$

显然这个变换核是可分离的，即

$$h(j,k;m,n)=h_1(j,m)h_2(k,n) \tag{4.5-32}$$

二维 HWHT 正、反变换表示式为

$$F(j,k)=\frac{1}{\sqrt{MN}}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f(m,n)(-1)^{\sum_{i=0}^{l-1}[b_i(j)b_i(m)+b_i(k)b_i(n)]} \tag{4.5-33}$$

$$f(m,n)=\frac{1}{\sqrt{MN}}\sum_{j=0}^{M-1}\sum_{k=0}^{N-1} F(j,k)(-1)^{\sum_{i=0}^{l-1}[b_i(j)b_i(m)+b_i(k)b_i(n)]} \tag{4.5-34}$$

由于变换核可分离，又由于$\boldsymbol{H}_N$的对称性，二维变换的矩阵形式是

$$\boldsymbol{F}=\boldsymbol{H}_M\boldsymbol{f}\boldsymbol{H}_N \tag{4.5-35}$$

$$\boldsymbol{f}=\boldsymbol{H}_M\boldsymbol{F}\boldsymbol{H}_N \tag{4.5-36}$$

可见正反变换不仅相同，而且有十分简单的形式。

4.5-4　快速 Walsh-Hadamard 变换

Walsh-Hadamard 变换也可以用快速算法来实现。其中一种算法的基础与快速傅立叶变换

的蝶形运算雷同。由于 Walsh-Hadamard 的变换核是可分离的，二维的变换可以由二次一维变换来实现，所以下面只讨论一维的快速算法。

● 快速 WWHT 变换

下面以 4 点的变换来说明快速算法。此时 $N=4$，故由 (4.5-7) 式可得

$$F(k)=\frac{1}{2}\sum_{n=0}^{3}f(n)(-1)^{\sum_{i=0}^{1}b_i(k)b_i(n)} \tag{4.5-37}$$

$$F(0)=\frac{1}{2}[f(0)+f(1)+f(2)+f(3)] \tag{4.5-38a}$$

$$F(1)=\frac{1}{2}[f(0)+f(1)-f(2)-f(3)] \tag{4.5-38b}$$

$$F(2)=\frac{1}{2}[f(0)-f(1)+f(2)-f(3)] \tag{4.5-38c}$$

$$F(3)=\frac{1}{2}[f(0)-f(1)-f(2)+f(3)] \tag{4.5-38d}$$

若定义

$$F_0=\frac{1}{2}[f(0)+f(2)],\quad F_1=\frac{1}{2}[f(1)+f(3)] \tag{4.5-39a}$$

$$F_2=\frac{1}{2}[f(0)-f(2)],\quad F_3=\frac{1}{2}[f(1)-f(3)] \tag{4.5-39b}$$

则 (4.5-38) 式可改写成

$$F(0)=F_0+F_1,\quad F(1)=F_2+F_3 \tag{4.5-40a}$$

$$F(2)=F_0-F_1,\quad F(3)=F_2-F_3 \tag{4.5-40b}$$

因此整个计算过程由图 4.5-3 所示的信号流程图来表示。由图中可看出与快速傅立叶变换一样，具有蝶形运算的基本结构。

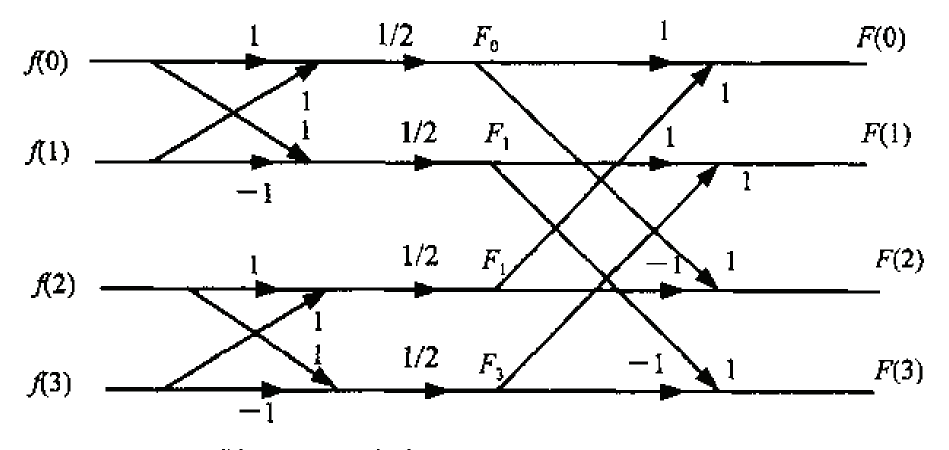


图 4.5-3　4 点快速 WWHT 变换信号流程图

● **快速 HWHT 变换**

因为 HWHT 变换与 WWHT 变换的不同仅在变换矩阵中元素的顺序不同，因此这种快速算法只需对上一小节讨论的快速算法稍加修改就可得到。最简单的方法就是将图 4.5-3 输出端的结果拉出，改变成 HWHT 所需的顺序即可。

4.6 Haar 变换

Haar 变换是以 Haar 函数为其基底的对称可分离幺正变换。它的大小限制为 $N=2^l$，其中 l 为正整数。傅立叶变换的基底函数仅在频率域上不同，而 Haar 变换却在位置 (position) 及尺度 (scale) 上产生不同，此点由其基底函数 (例如 4.6-4 式) 可明显看出，这样的特征使 Haar 变换与先前介绍过的其他变换法有所区别。事实上，Haar 变换被视为 4.11 节要介绍的小波变换 (wavelet transform) 的简单例子。

因为 Haar 函数在尺度及位置上产生变化，所以必须用一对偶指标技术来表示。令整数 k，$0 \leqslant k \leqslant N-1$，以另两个整数 p 与 q 来表示

$$k = 2^p + q - 1 \tag{4.6-1}$$

值得注意的是：不仅 k 为 p 与 q 的函数，p 与 q 亦为 k 的函数。对任意 $k>0$，2^p 为 $2^p \leqslant k$ 的 2 的最大乘幂，而 $q-1$ 为余数。例如 $N=4$ 时 p 与 q 的值如下：

k	0	1	2	3
p q	0 0	0 1	1 1	1 2

在 [0,1] 区间中，Haar 函数定义为

$$h_0(x) = \frac{1}{\sqrt{N}} \tag{4.6-2}$$

$$h_k(x) = \frac{1}{\sqrt{N}} \begin{cases} 2^{\frac{p}{2}}, & \dfrac{q-1}{2^p} \leqslant x < \dfrac{q-\frac{1}{2}}{2^p} \\ -2^{\frac{p}{2}}, & \dfrac{q-\frac{1}{2}}{2^p} \leqslant x < \dfrac{q}{2^p} \\ 0, & \text{其他情况} \end{cases} \tag{4.6-3}$$

若令 $x=i/N$，$i=0,1,\cdots,N-1$，则可造出一组基底函数。例如 8×8 的 Haar 幺正变换核的矩阵如 (4.6-4) 式所示。

$$
\boldsymbol{Hr}_8=\frac{1}{\sqrt{8}}\begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1\\
1 & 1 & 1 & 1 & -1 & -1 & -1 & -1\\
\sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2}\\
2 & -2 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 2 & -2 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 2 & -2 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 2 & -2
\end{bmatrix} \tag{4.6-4}
$$

由于矩阵中存在许多常数与零值，所以 Haar 变换的运算非常快速。由变换矩阵的列中可看出、处理输入数据的解析度可由粗到细，每次以 2 的乘幂变化。在图像处理的应用中、Haar 变换适合用来找边缘之类的特征，这是因为其变换基底函数中就存在粗细不同的特征．对输入数据作变换、相当于找出其与基底函数间的相似性。

4.7 斜变换

对于 $N\times N$ 的斜变换 (Slant transform) 我们定义为

$$
\boldsymbol{S}_N=\frac{1}{\sqrt{2}}\left[\begin{array}{cc|c|cc|c}
1 & 0 & & 1 & 0 & \\
a_N & b_N & \boldsymbol{0} & -a_N & b_N & \boldsymbol{0}\\
\hline
\multicolumn{2}{c|}{\boldsymbol{0}} & \boldsymbol{I}_{(N/2)-2} & \multicolumn{2}{c|}{\boldsymbol{0}} & \boldsymbol{I}_{(N/2)-2}\\
\hline
0 & 1 & & 0 & -1 & \\
-b_N & a_N & \boldsymbol{0} & b_N & a_N & \boldsymbol{0}\\
\hline
\multicolumn{2}{c|}{\boldsymbol{0}} & \boldsymbol{I}_{(N/2)-2} & \multicolumn{2}{c|}{\boldsymbol{0}} & -\boldsymbol{I}_{(N/2)-2}
\end{array}\right]
\left[\begin{array}{c|c}
\boldsymbol{S}_{N/2} & \boldsymbol{0}\\
\hline
\boldsymbol{0} & \boldsymbol{S}_{N/2}
\end{array}\right] \tag{4.7-1}
$$

其中 $\boldsymbol{I}_M$ 表示一个 $M\times M$ 的单位矩阵而且

$$
\boldsymbol{S}_2=\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix} \tag{4.7-2}
$$

另外，参数 a_N 和 b_N 的定义为

$$
a_N=\left(\frac{3N^2}{4(N^2-1)}\right)^{\frac{1}{2}},\quad b_N=\left(\frac{N^2-4}{4(N^2-1)}\right)^{\frac{1}{2}} \tag{4.7-3}
$$

使用 (4.7-1) 式和 (4.7-3) 式的方法、可将一个 4×4 的斜变换矩阵写成

$$
\boldsymbol{S}_4=\frac{1}{\sqrt{2}}\begin{bmatrix}
1 & 0 & 1 & 0\\
a_4 & b_4 & -a_4 & b_4\\
0 & 1 & 0 & -1\\
-b_4 & a_4 & b_4 & a_4
\end{bmatrix}\cdot\frac{1}{\sqrt{2}}\begin{bmatrix}
1 & 1 & 0 & 0\\
1 & -1 & 0 & 0\\
0 & 0 & 1 & 1\\
0 & 0 & 1 & -1
\end{bmatrix},\quad a_4=\frac{2}{\sqrt{5}},b_4=\frac{1}{\sqrt{5}} \tag{4.7-4}
$$

$$\therefore \boldsymbol{S}_4 = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ \frac{3}{\sqrt{5}} & \frac{1}{\sqrt{5}} & \frac{-1}{\sqrt{5}} & \frac{-3}{\sqrt{5}} \\ 1 & -1 & -1 & -1 \\ \frac{1}{\sqrt{5}} & \frac{-3}{\sqrt{5}} & \frac{3}{\sqrt{5}} & \frac{-1}{\sqrt{5}} \end{bmatrix} \tag{4.7-5}$$

- **斜变换的性质**

1. 斜变换必须是实数且正交，因此

$$\boldsymbol{S} = \boldsymbol{S}^*, \quad \boldsymbol{S}^{-1} = \boldsymbol{S}^{\mathrm{T}} \tag{4.7-6}$$

2. 斜变换是一种快速变换，其对一个 N 维向量的计算量为 $O(N\log_2 N)$ 的等级。
3. 它具有好的能量集中特性。

4.8 KL 变换

由 Karhunen 和 L'oeve 两人所共同提出的 KL 变换，原先是作为对连续随机过程的级数展开之用。而对于随机序列，Hotelling 最先探讨主值成分法 (the method of principal components)，其实它是 KL 级数展开的等效离散版本。因而有时 KL 变换也称为 Hotelling 变换，或是主值成分法。

考虑一维么正矩阵 $\boldsymbol{\Phi}$，如下所示

$$\boldsymbol{Y} = \boldsymbol{\Phi}^*\boldsymbol{X}, \quad \boldsymbol{X} = \boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{Y} \tag{4.8-1}$$

此处 $\boldsymbol{X} = [x_0, x_1, x_2, \cdots, x_{N-1}]^{\mathrm{T}}$ 和 $\boldsymbol{Y} = [y_0, y_1, y_2, \cdots, y_{N-1}]^{\mathrm{T}}$ 分别为在空间域及变换域中的数据。空间域中的数据为实数、变换域中的数据为复数。令规范正交序列 $\phi_r(k)$ 为变换矩阵 $\boldsymbol{\Phi} = \{\phi(r,k)\}$ 的行。同时令 $\boldsymbol{\Phi}_r$ 为一表示基底序列 $\{\phi_r(k)\}$ 的列向量，也就是 $[\phi_r(0), \phi_r(1), \cdots, \phi_r(N-1)]^{\mathrm{T}}$，则 $\boldsymbol{X}$ 可以被写成这些基底向量的权重和，亦即

$$\mathbf{X} = \begin{bmatrix} \phi_0(0) & \phi_1(0) & \cdots & \phi_{N-1}(0) \\ \phi_0(1) & \phi_1(1) & \cdots & \phi_{N-1}(1) \\ \vdots & \vdots & \cdots & \vdots \\ \phi_0(N-1) & \phi_1(N-1) & \cdots & \phi_{N-1}(N-1) \end{bmatrix}\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{N-1} \end{bmatrix} \tag{4.8-2}$$

$$= [\boldsymbol{\Phi}_0 \boldsymbol{\Phi}_1 \boldsymbol{\Phi}_2 \cdots \boldsymbol{\Phi}_{N-1}]\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{N-1} \end{bmatrix} = \sum_{r=0}^{N-1} y_r \boldsymbol{\Phi}_r$$

令 $\boldsymbol{X}$ 的截取表示式 (truncated representation) 为 $\boldsymbol{X}_L$

$$\boldsymbol{X}_L=\sum_{r=0}^{L-1} y_r \boldsymbol{\Phi}_r \tag{4.8-3}$$

(4.8-3) 式中的截取表示式所导致的均方值误差为

$$\begin{aligned}\varepsilon &= E[(\boldsymbol{X}-\boldsymbol{X}_L)^{\mathrm{T}}(\boldsymbol{X}-\boldsymbol{X}_L)] \\ &= E\left[\left(\sum_{r=L}^{N-1} y_r \boldsymbol{\Phi}_r\right)^{\mathrm{T}}\left(\sum_{r=L}^{N-1} y_r \boldsymbol{\Phi}_r\right)\right]=\sum_{r=L}^{N-1} E\left[y_r^2\right]\end{aligned} \tag{4.8-4}$$

此处设 $\boldsymbol{X}$ 为一实数随机变量且其平均值为零。因为 $y_r=\boldsymbol{\Phi}_r^{\mathrm{T}}\boldsymbol{X}$，所以

$$\varepsilon=\sum_{r=L}^{N-1} E[\boldsymbol{\Phi}_r^{\mathrm{T}}(\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}})\boldsymbol{\Phi}_r]=\sum_{r=L}^{N-1}\boldsymbol{\Phi}_r^{\mathrm{T}}\boldsymbol{R}_{XX}\boldsymbol{\Phi}_r \tag{4.8-5}$$

此处 $\boldsymbol{R}_{XX}$ 为向量 $\boldsymbol{X}$ 的协方差矩阵。

为得到最佳变换，我们要找出符合规范正交性限制 (orthonormality constraint) $\boldsymbol{\Phi}_r^{\mathrm{T}}\boldsymbol{\Phi}_s=\delta_{r-s}$，并可将 ε 最小化的基底函数 $\boldsymbol{\Phi}_r$。

用拉格朗日乘数法，将下式最小化

$$J=\sum_{r=L}^{N-1}[\boldsymbol{\Phi}_r^{\mathrm{T}}\boldsymbol{R}_{XX}\boldsymbol{\Phi}_r-\lambda_r(\boldsymbol{\Phi}_r^{\mathrm{T}}\boldsymbol{\Phi}_r-1)] \tag{4.8-6}$$

将上式有关 $\boldsymbol{\Phi}_r$ 的每一项取梯度并令其为零向量可得

$$2\boldsymbol{R}_{XX}\boldsymbol{\Phi}_r-2\lambda_r\boldsymbol{\Phi}_r=\boldsymbol{0}\Rightarrow\boldsymbol{R}_{XX}\boldsymbol{\Phi}_r=\lambda_r\boldsymbol{\Phi}_r \tag{4.8-7}$$

也就是说

$$\boldsymbol{R}_{XX}\boldsymbol{\Phi}^{\mathrm{T}}=\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Lambda} \tag{4.8-8}$$

此处 $\boldsymbol{\Lambda}=\mathrm{diag}(\lambda_0,\lambda_1,\cdots,\lambda_{N-1})$，即主对角元素为 $\lambda_0,\lambda_1,\cdots,\lambda_{N-1}$ 的对角矩阵。因此 $\boldsymbol{\Phi}_r$ 为一协方差矩阵 $\boldsymbol{R}_{XX}$ 的特征向量，λ_r 为特征多项式 $|\lambda_r\boldsymbol{I}-\boldsymbol{R}_{XX}|$ 的一个根。由于 $\boldsymbol{R}_{XX}$ 为一实数对称矩阵，所有 $\{\lambda_r\}$ 为相异的非负值实数，所以最小的 ε 为

$$\varepsilon_{\min}=\sum_{r=L}^{N-1}\boldsymbol{\Phi}_r^{\mathrm{T}}(\lambda_r\boldsymbol{\Phi}_r)=\sum_{r=L}^{N-1}\lambda_r \tag{4.8-9}$$

如此，ε 由将 λ_r 降阶排列来达成最小化。基底向量集合将会把协方差矩阵 $\boldsymbol{R}_{XX}$ 对角化，这可由下式看出

$$\boldsymbol{R}_{YY}=\boldsymbol{\Phi}\boldsymbol{R}_{XX}\boldsymbol{\Phi}^{\mathrm{T}}=\boldsymbol{\Phi}\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Lambda}=\boldsymbol{\Lambda} \tag{4.8-10}$$

总之，$\boldsymbol{\Phi}$ 为 KLT 的么正矩阵，它可产生一对角矩阵 $\boldsymbol{R}_{YY}$，因而可以将变换后的系数间的相关性完全去除。此外，它可将总信号能量重新配置给前 L 个系数。

比较 (4.8-4) 与 (4.8-9) 式，我们得到

$$\lambda_r=E\left[y_r^2\right] \tag{4.8-11}$$

既然 ε 是由特征值 λ_r 的降阶排列来达成最小化，ε 也可由将系数 y_r 的振幅降阶排列来达成最小化。值得注意的是，虽然许多矩阵都可以去除输入信号的相关性，但 KLT 不只可以完美地

去除输入信号的相关性，同时可以达成信号能量重新配置的最佳化。不过，显然基底函数与协方差矩阵 $\boldsymbol{R}_{XX}$ 有关，因此不能事先决定。此外，给一协方差矩阵，找出 KLT 的基底函数通常需要大量的计算。基于上述理由，KLT 实际上很少用。反倒是像 DCT 等与信号数据无关的次佳变换常被使用。

4.9 哈特莱变换

Bracewall 于 1983 年提出一个离散实数值么正变换以取代傅立叶变换。称作哈特莱变换 (Hartley transform) 是因为此变换是从 1942 年哈特莱所提出的连续积分版本中推导出 哈特莱变换是利用以下的正交函数 cas(θ) 当基底函数而做正交变换

$$\mathrm{cas}(\theta) = \sin(\theta) + \cos(\theta) \tag{4.9-1}$$

二维离散哈特莱变换对定义为

$$F(j,k) = \frac{1}{N}\sum_{m=0}^{N-1}\sum_{n=0}^{N-1} f(m,n)\,\mathrm{cas}\left[\frac{2\pi}{N}(mj+nk)\right] \tag{4.9-2}$$

$$f(m,n) = \frac{1}{N}\sum_{j=0}^{N-1}\sum_{k=0}^{N-1} F(j,k)\,\mathrm{cas}\left[\frac{2\pi}{N}(mj+nk)\right] \tag{4.9-3}$$

哈特莱变换与 DFT 有类似的性质。对某种应用而言，哈莱特变换比 DFT 在计算上有效率，但有些时候却比较费时。

4.10 SVD 变换

一个 $N\times N$ 矩阵 $\boldsymbol{A}$ 可表示成

$$\boldsymbol{A} = \boldsymbol{U}\Lambda\boldsymbol{V}^{\mathrm{T}} \tag{4.10-1}$$

其中 $\boldsymbol{U}$ 与 $\boldsymbol{V}$ 的规范正交行向量分别为 $\boldsymbol{AA}^{\mathrm{T}}$ 与 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ 的特征向量，Λ 为一对角元素是 $\boldsymbol{A}$ 的特征值的 $N\times N$ 对角矩阵。此特征值又称奇异值 (singular value)。因 $\boldsymbol{U}$ 与 $\boldsymbol{V}$ 为么正矩阵，故可推得

$$\Lambda = \boldsymbol{U}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V} \tag{4.10-2}$$

上式为正变换，(4.10-1) 式为反变换，二者为变换对，此变换称为奇异值分解 (singular value decomposition, SVD) 变换。如果 **A** 为对称，则 $\boldsymbol{U}$ 就等于 $\boldsymbol{V}$。此外，$\boldsymbol{V}$ 视变换图像 $\boldsymbol{A}$ 而定，一般来说在变换过程中必需计算每一图像的 $\boldsymbol{AA}^{\mathrm{T}}$ 与 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ 的特征向量。另外值得注意的是：因为 Λ 为一对角矩阵，它至多有 N 个非零元素，由此可得至少 N 倍的无失真压缩 通常，部分奇异值都小到可以忽略，因此失真压缩可借助忽略较小奇异值来达成，此时均方误差就是所忽略的奇异值的总和。

SVD 变换看似不可思议的压缩功能可能会产生误解，虽然全幅图像可以压缩成 Λ 的对角元素，但图像的核心矩阵 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 必须由原图像计算取得，而且它们必须在接收端重建图像前送达。

在此举一实例说明 SVD。考虑矩阵 $\boldsymbol{A}$ 如下

$$\mathbf{A}=\begin{bmatrix}1 & 2 & 1\\ 2 & 3 & 2\\ 1 & 2 & 1\end{bmatrix} \tag{4.10-3}$$

因为 $\boldsymbol{A}$ 为对称矩阵，故可知 $\boldsymbol{U}=\boldsymbol{V}$，因此

$$\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}=\begin{bmatrix}6 & 10 & 6\\ 10 & 17 & 10\\ 6 & 10 & 6\end{bmatrix} \tag{4.10-4}$$

其特征值为

$$\begin{bmatrix}\lambda_1\\ \lambda_2\\ \lambda_3\end{bmatrix}=\begin{bmatrix}28.86\\ 0.14\\ 0\end{bmatrix} \tag{4.10-5}$$

特征向量则为

$$\boldsymbol{U}_1=\boldsymbol{V}_1=\begin{bmatrix}0.454\\ 0.766\\ 0.454\end{bmatrix},\quad \boldsymbol{U}_2=\boldsymbol{V}_2=\begin{bmatrix}0.542\\ -0.643\\ 0.542\end{bmatrix},\quad \boldsymbol{U}_3=\boldsymbol{V}_3=\begin{bmatrix}-0.707\\ 0\\ -0.707\end{bmatrix} \tag{4.10-6}$$

奇异值在下列矩阵的主对角线上

$$\boldsymbol{\Lambda}=\boldsymbol{U}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V}=\begin{bmatrix}5.37 & 0 & 0\\ 0 & -0.372 & 0\\ 0 & 0 & 0\end{bmatrix} \tag{4.10-7}$$

SVD 的展开式可写成

$$\boldsymbol{A}=\sum_{j=1}^{3}\boldsymbol{\Lambda}_{j,j}\boldsymbol{U}_j\boldsymbol{V}_j^{\mathrm{T}} \tag{4.10-8}$$

注意到第二个奇异值远小于第一个，因此可忽略 (4.10-8) 式中的 $\Lambda_{2,2}$，而得 $\boldsymbol{A}$ 的近似值如下

$$\boldsymbol{A}\approx\boldsymbol{\Lambda}_{1,1}\boldsymbol{U}_1\boldsymbol{V}_1^{\mathrm{T}}=\begin{bmatrix}1.11 & 1.87 & 1.11\\ 1.87 & 3.15 & 1.87\\ 1.11 & 1.87 & 1.11\end{bmatrix} \tag{4.10-9}$$

4.11　小波变换

近年来有一个新变换常用在图像压缩、边缘与特征检测以及纹理 (texture) 分析等问题上，这个变换就是小波变换 (wavelet transform, WT)。与傅立叶系列的变换所采用的弦式波基底相比较而言，小波变换的基底有较短的持续时间 (time duration)，因而得名。图 4.11-1 显示二个频率不同的余弦波形及二个位置与频率皆不同的典型小波图形。

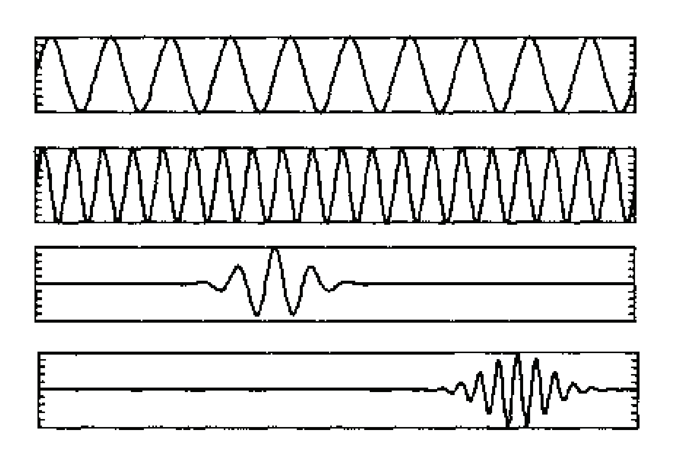


图 4.11-1　弦式波与小波

一个暂态 (transient) 信号的分量通常只在很短的时间区间内不为零，同理许多图像的重要特征 (如边缘等) 也只出现在局部的小空间上。这些分量与任何傅立叶系列的基底函数都不像，因此表达这些信号分量时显得非常没有效率。小波的出现正是针对这个缺点而来。

4.11-1　时频分解

假想有一个较平滑 (频率较低) 的信号加上在某个时间所产生短而快速的暂态信号，称此合成信号为 $f(t)$。以传统傅立叶分析

$$F(\Omega)=\int_{-\infty}^{\infty}f(t)\mathrm{e}^{-\mathrm{j}\Omega t}\mathrm{d}t \tag{4.11-1}$$

我们从频域上可知有暂态信号出现，但却不知在何时出现。如果在取傅立叶变换前先乘上一个时窗，则至少可知暂态发生在哪一个时窗范围内。设此时窗函数为 $g(t)$，则此时窗函数的傅立叶变换，或称短时傅立叶变换 (short time Fourier transform, STFT)，可写成

$$F(\Omega,\tau)=\int_{-\infty}^{\infty}f(t)g^{*}(t-\tau)\mathrm{e}^{-\mathrm{j}\Omega t}\mathrm{d}t \tag{4.11-2}$$

当时窗 $g(t)$ 为高斯函数时，STFT 就称为 Gabor 变换。STFT 有固定持续时间的时窗 $g(t)$，以及一个固定的频率解析度。这是测不准原理 (uncertainty principle)的自然结果。该原理是说，对任何变换对 $g(t)\leftrightarrow G(\Omega)$，

$$\sigma_T \sigma_\Omega \geqslant \frac{1}{2} \tag{4.11-3}$$

其中只有当 $g(t)$ 为高斯函数时等号才成立，而 σ_T 与 σ_Ω 分别为度量 $g(t)$ 及 $G(\Omega)$ 的均方根散开程度

$$\sigma_T^2 = \frac{\int t^2 |g(t)|^2 \mathrm{d}t}{\int |g(t)|^2 \mathrm{d}t}, \quad \sigma_\Omega^2 = \frac{\int \Omega^2 |G(\Omega)|^2 \mathrm{d}\Omega}{\int |G(\Omega)|^2 \mathrm{d}\Omega} \tag{4.11-4}$$

图 4.11-2 显示 STFT 在时频域平面上的解析度。

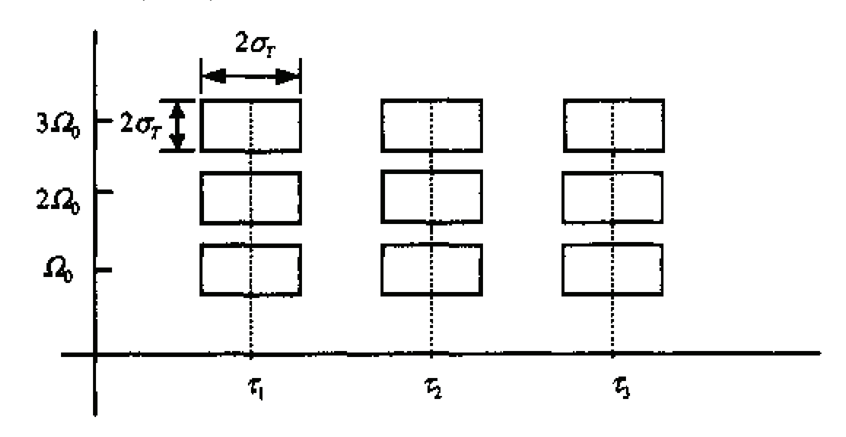


图 4.11-2　STFT 在时频域平面上的解析度

小波变换的基底是由一个原型 (prototype)函数的缩张 (dilation)与平移 (translation) 所形成的。这些基底函数具有短持续时间且高频率，以及长持续时间且低频率的特性，因此相当适合表达高频的突发暂态信号或是长时间缓慢变化的信号。其时频域上的解析度如图 4.11-3 所示。比较图 4.11-2 与 4.11-3 可看出，小波变换提供一个很有弹性的时频解析度。

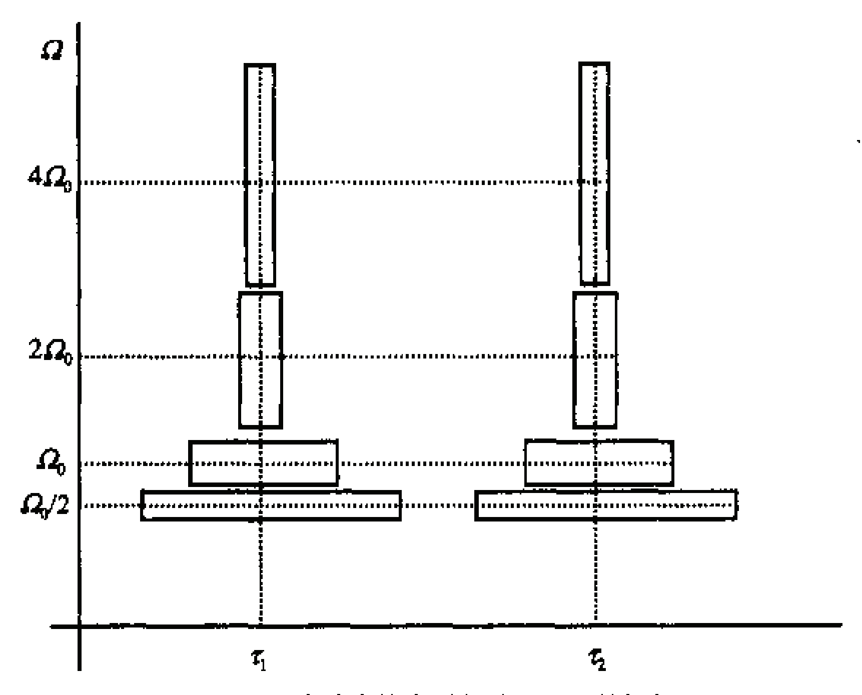


图 4.11-3　小波变换在时频域平面上的解析度

4.11-2　连续小波变换

与短时傅立叶变换相似的是：小波变换将一个时间函数映射至一个 a 和 τ 的二维函数。此处参数 a 称为尺度 (scale) 参数，作用是将函数做压缩或伸展；而 τ 为沿着时间轴的小波函数的移动。设信号 $f(t)$ 为平方可积 (square integrable)，表示为 $f(t)\in L^2(R)$，也就是

$$\int f^2(t)\mathrm{d}t < \infty \tag{4.11-5}$$

则 $f(t)$ 的连续小波变换 (continuous WT, CWT) 为

$$\mathrm{CWT}(a,\tau)=\frac{1}{\sqrt{a}}\int f(t)\psi\left(\frac{t-\tau}{a}\right)\mathrm{d}t \tag{4.11-6}$$

此处 $\psi(t)$ 为基本 (basic)或母小波 (mother wavelet)且 $\psi((t-\tau)/a)/\sqrt{a}$ 为小波基底函数，有时也称子小波 (baby wavelet)。若满足下列许可条件 (admissibility condition)

$$\Psi(0)=\int_{-\infty}^{\infty}\psi(t)\,\mathrm{d}t \tag{4.11-7}$$

则小波变换为可逆的 (invertible)。

基本小波可以为实数或复数，这也会导致变换的结果对应为实数或复数。当 $\psi(t)$ 为复数时，(4.11-6) 及 (4.11-7) 式中将改采用其共轭复数。在某些应用中，由于小波变换的相角含有许多有用的信息，所以使用复数小波会较有益处。以下是一些 $\psi(t)$ 及其傅立叶变换的例子：

(1) Morlet

$$\begin{aligned}\psi(t)&=\mathrm{e}^{j\omega_0 t}\mathrm{e}^{-\frac{t^2}{2}}\\ \Psi(\omega)&=\sqrt{2\pi}\mathrm{e}^{-\frac{(\omega-\omega_0)^2}{2}}\end{aligned} \tag{4.11-8}$$

(2) 高斯二次导数

$$\begin{aligned}\psi(t)&=(1-t^2)\mathrm{e}^{-\frac{t^2}{2}}\\ \Psi(\omega)&=\sqrt{2\pi}\omega^2\mathrm{e}^{-\frac{\omega^2}{2}}\end{aligned} \tag{4.11-9}$$

(3) Haar

$$\begin{aligned}\psi(t)&=\begin{cases}1, & 0\leqslant t\leqslant \dfrac{1}{2}\\ -1, & \dfrac{1}{2}\leqslant t<1\\ 0, & \text{其他情况}\end{cases}\\ \Psi(\omega)&=j\mathrm{e}^{-\mathrm{i}\frac{\omega}{2}}\frac{\sin^2(\omega/4)}{\omega/4}\end{aligned} \tag{4.11-10}$$

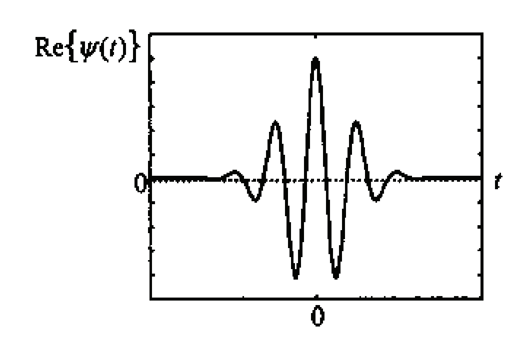
(4) Shannon

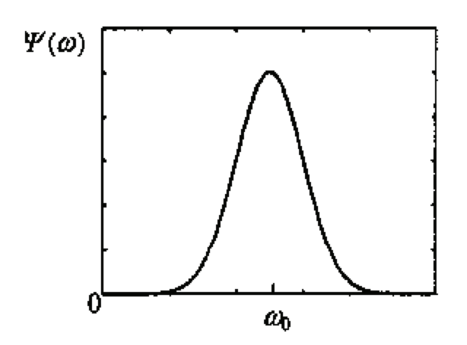
$$\psi(t)=\frac{\sin(\pi t/2)}{\pi t/2}\cos\left(\frac{3\pi t}{2}\right)$$

$$\Psi(\omega)=\begin{cases}1, & \pi<|\omega|<2\pi\\ 0, & \text{其他情况}\end{cases} \tag{4.11-11}$$

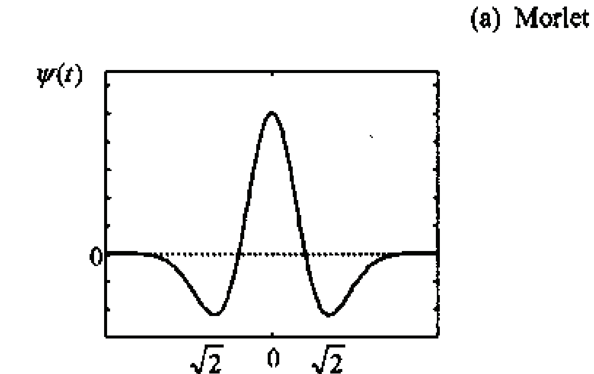
图 4.11-4 显示上述小波及其傅立叶变换。图 4.11-5 则为小波 Haar 及其子小波。从图 4.11-4 及 4.11-5 我们可以推论出下列小波的特性：

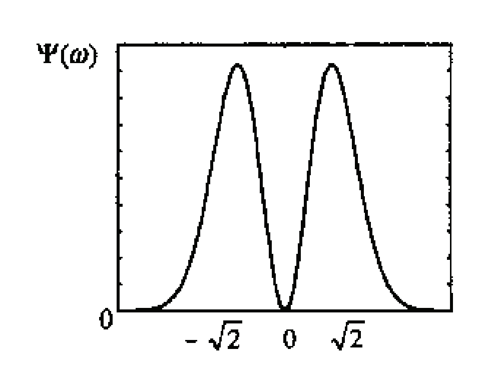
1. 当 $\omega=0$ 时 $\Psi(\omega)=0$，也就是 $\int\psi(t)\,\mathrm{d}t=0$，换句话说：它们具备零 DC 值。
2. 它们皆为带通信号。
3. 它们随时间迅速朝零衰减。



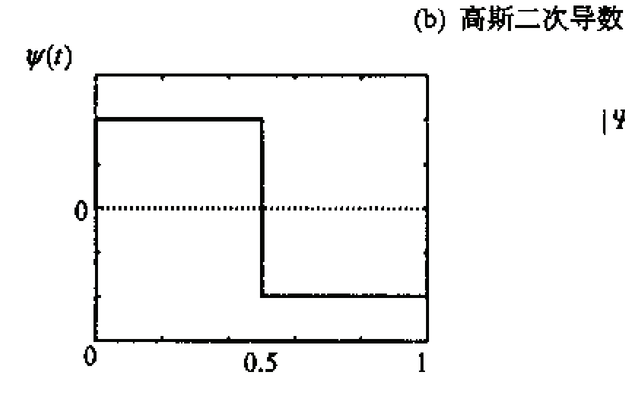


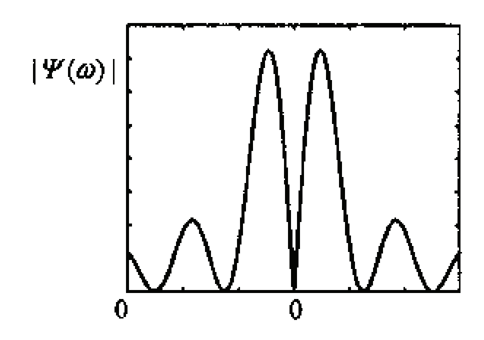
(a) Morlet





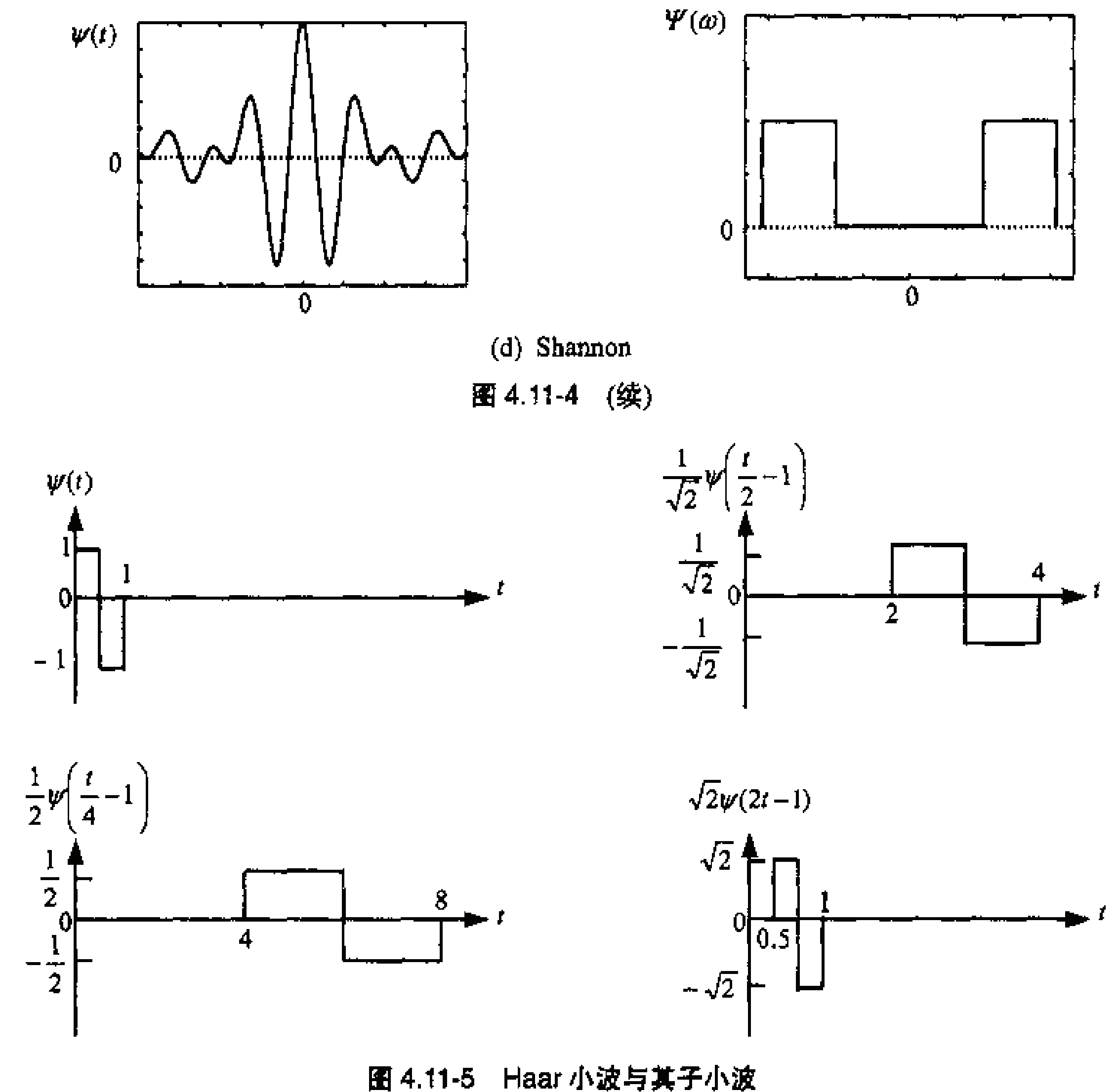
(b) 高斯二次导数





(c) Haar

图 4.11-4　一些小波及其傅立叶变换 (待续)



(d) Shannon

图 4.11-4　(续)

图 4.11-5　Haar 小波与其子小波

4.11-3　离散小波变换

在 (4.11-6) 式中，(a,τ) 皆为连续变量且在 $f(t)$ 的 CWT 描述中具有冗余性 (redundancy)。因此没有必要针对所有可能的 (a,τ) 值去计算 $\mathrm{CWT}(a,\tau)$。此外，针对有限 (a,τ) 值的计算就有实际上的必要。

当 (a,τ) 为离散值时，若令 $a=a_0^m$，$\tau=n\tau_0a_0^m$，其中 m 与 n 为整数，则此时离散小波变换为

$$\mathrm{DWT}(m,n)=\int f(t)\psi_{mn}(t)\mathrm{d}t \tag{4.11-12}$$

此处

$$\psi_{mn}(t)=a_0^{-\frac{m}{2}}\psi(a_0^{-m}t-n\tau_0),\quad \psi_{00}(t)=\psi(t) \tag{4.11-13}$$

对于一个规范正交的小波 $\{\psi_{mn}(t)\}$ 而言，它必须满足

$$\int \psi_{mn}(t)\psi_{m'n'}(t)\mathrm{d}t=\begin{cases}1, & m=m', n=n' \\ 0, & \text{其他情况}\end{cases} \tag{4.11-14}$$

这表示在同一个尺度 m 或跨尺度下，它都是规范正交的。

在许多工程应用上，常采用$a_0=2, \tau_0=1$，亦即用 2 次幂格点 (dyadic grid) 取样。针对此状况，接下来我们将以*多重解析度* (multiresolution) 的观点将小波变换与*子带滤波* (subband filter bank) 相连结，进而可以用滤波器来展现小波变换。

当函数 f 属于一个空间 $L^2(R)$，则有一*尺度* (scaling) 函数 $\phi(t)$ 使得 f 从 $L^2(R)$ 空间滤得较为平滑的子信号。所谓尺度函数可视为低通滤波器，而同一层的高通滤波器即为小波函数 $\psi(t)$。前述的平滑子信号再由经缩张和位移后的尺度函数进一步析出细节部分而逐渐平滑化，依序进行使原始信号被映射到一连串的子空间。设此一连串的子空间以 V_m，$m \in Z$ (整数) 表示且此子空间满足下列性质：

1. 包含性：$\cdots V_2 \subset V_1 \subset V_0 \subset V_{-1} \subset V_{-2} \cdots$

   ←粗 (解析度低)　细 (解析度高)→

2. 完备性：$\bigcap_{m\in Z} V_m=\{0\}$，$\bigcup_{m\in Z} V_m=L^2(R)$

3. 尺度性：$f(t)\in V_m \Leftrightarrow f(2t)\in V_{m-1}$

4. 基底性质 对于一个尺度函数$\phi(t)\in V_0$使得$\phi_{mn}(t)=2^{-\frac{m}{2}}\phi(2^{-m}t-n)$所构成的集合是$V_m$的一个规范正交基底，也就是$\int_{-\infty}^{\infty}\phi_{mn}(t)\phi_{mn'}(t)=\delta_{n-n'}$。

设 V_{m-1} 空间经由 $\phi(t)$ 滤除的信号存在另一正交基底组成的子空间 W_m 中使得 $V_{m-1}=V_m\oplus W_m$，$V_m \perp W_m$。如同尺度函数$\phi(t)$可*展延* (span) 出 V 信号空间，另一函数$\psi(t)$则展延出 W 信号空间，而$\psi(t)$就是先前提过的小波函数。接着我们以向量投影的观念讨论信号空间的分解。设投影运算子P_m及Q_m将$P_{m-1}f$ 分别投影到V_m与W_m空间，即$P_{m-1}f=P_mf+Q_mf$，其中P_mf为低通部分而Q_mf 为高频细微信号部分且$P_mf\in V_m, Q_mf\in W_m$。

设尺度函数$\phi(t)$的位移集合$\{\phi(t-n)\}$展延出V_0空间，则集合$\{\phi(2t-n)\}$展延出V_{-1}空间。因此，由包含性知$\phi(t)$可表示成$\phi(2t-n)$的线性组合，或

$$\phi(t)=2\sum_n h_0(n)\phi(2t-n) \tag{4.11-15}$$

同理小波函数也可写成

$$\psi(t)=2\sum_n h_1(n)\phi(2t-n) \tag{4.11-16}$$

其中 $h_0(n)$ 与 $h_1(n)$ 为*内部尺度基底系数* (interscale basis coefficients)

$$h_0(n)=\frac{1}{2}\int\phi\left(\frac{1}{2}\right)\phi(t-n)\mathrm{d}t \tag{4.11-17a}$$

$$h_1(n)=\frac{1}{2}\int\psi\left(\frac{1}{2}\right)\phi(t-n)\mathrm{d}t \tag{4.11-17b}$$

当信号f在V_0空间上，且集合$\{\phi(t-n)\}$展延出V_0空间，则f可表示为

$$f(t)=\sum_n c_{0,n}\phi(t-n)=\sum_n c_{0,n}\phi_{0,n}(t) \tag{4.11-18}$$

其中

$$c_{0,n}=\langle f,\phi_{0,n}\rangle=\int f(t)\phi(t-n)\mathrm{d}t \tag{4.11-19}$$

而n为信号序列数。信号投影可用子空间基底的线性组合表示

$$f(t)=P_1 f+Q_1 f=f_v^1+f_w^1=\sum_n c_{1,n}\phi_{1,n}(t)+\sum_n d_{1,n}\psi_{1,n}(t) \tag{4.11-20}$$

其中$\psi_{mn}(t)=2^{-\frac{m}{2}}\psi(2^{-m}t-n)$。因此尺度系数$c_{1,n}$和小波系数$d_{1,n}$为

$$c_{1,n}=\langle f_v^1,\phi_{1,n}\rangle=\frac{1}{\sqrt{2}}\int f_v^1\phi\left(\frac{t}{2}-n\right)\mathrm{d}t \tag{4.11-21}$$

$$d_{1,n}=\langle f_w^1,\psi_{1,n}\rangle=\frac{1}{\sqrt{2}}\int f_w^1(t)\psi\left(\frac{t}{2}-n\right)\mathrm{d}t \tag{4.11-22}$$

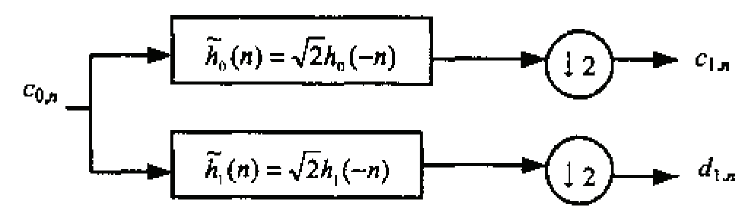
因为$f(t)=f_v^1(t)+f_w^1(t)$，所以$\langle f,\phi_{1,n}\rangle=\langle f_v^1,\phi_{1,n}\rangle+\langle f_w^1,\phi_{1,n}\rangle$，又$f_w^1$与$\phi_{1,n}$正交，故$\langle f_w^1,\phi_{1,n}\rangle=0$，因此$\langle f,\phi_{1,n}\rangle=\langle f_v^1,\phi_{1,n}\rangle=c_{1,n}$，则由 (4.11-21) 与 (4.11-15) 式可得

$$\begin{aligned}c_{1,n}&=\sqrt{2}\int f(t)\sum_k h_0(k)\phi(t-2n-k)\,\mathrm{d}t\\&=\sqrt{2}\sum_k h_0(k)\int f(t)\phi(t-2n-k)\,\mathrm{d}t=\sqrt{2}\sum_k h_0(k)\cdot c_{0,2n+k}\\&=\sqrt{2}\sum_k h_0(k-2n)\cdot c_{0,k}\end{aligned} \tag{4.11-23}$$

同理可得

$$d_{1,n}=\sqrt{2}\sum_k h_1(k-2n)\cdot c_{0,k} \tag{4.11-24}$$

$c_{1,n}$与$d_{1,n}$为信号f (可视为$c_{0,n}$) 做小波变换的第一层系数，尔后再以$c_{1,n}$为下一层的信号输入再行小波变换，依此类推即可得小波变换多重分解。由 (4.11-23) 及 (4.11-24) 式可知实现小波变换的演算方式应如图 4.11-6 所示，其中$\widetilde{h}_0(n)$和$\widetilde{h}_1(n)$即是分频滤波器组的系数。



注：圆形符号代表缩减取样，即每两点输入保留一个点当输出。

图 4.11-6 第一层多解析信号分解

现在考虑合成的部分。(4.11-17a) 式的内部尺度基底系数可推得如下。由 (4.11-15) 式可得

$$\phi\left(\frac{t}{2}\right)=2\sum_{k}h_0(k)\phi(t-k) \tag{4.11-25}$$

故

$$\begin{aligned}\int\phi\left(\frac{t}{2}\right)\phi(t-n)\mathrm{d}t&=2\int\sum_{k}h_0(k)\phi(t-k)\phi(t-n)\,\mathrm{d}t\\&=\sum_{k}2h_0(k)\int\phi(t-k)\phi(t-n)\,\mathrm{d}t\\&=\sum_{k}2h_0(k)\delta_{k-n}=2h_0(n)\end{aligned} \tag{4.11-26}$$

因此

$$h_0(n)=\frac{1}{2}\int\phi\left(\frac{t}{2}\right)\phi(t-n)\mathrm{d}t \tag{4.11-27}$$

同理由 (4.11-16) 式开始亦可得

$$h_1(n)=\frac{1}{2}\int\psi\left(\frac{t}{2}\right)\phi(t-n)\mathrm{d}t \tag{4.11-28}$$

系数 $c_{0,n}$ 可写成

$$c_{0,n}=\left\langle f,\phi_{0,n}\right\rangle=\left\langle f_v^1,\phi_{0,n}\right\rangle+\left\langle f_w^1,\phi_{0,n}\right\rangle \tag{4.11-29}$$

其中

$$\begin{aligned}\left\langle f_v^1,\phi_{0,n}\right\rangle&=\int f_v^1(t)\phi_{0,n}(t)\,\mathrm{d}t=\int\phi_{0,n}(t)\sum_{k}c_{1,k}\phi_{1,k}(t)\,\mathrm{d}t\\&=\sum_{k}c_{1,k}\frac{1}{\sqrt{2}}\int\phi(t-n)\phi\left(\frac{t}{2}-k\right)\mathrm{d}t\end{aligned} \tag{4.11-30}$$

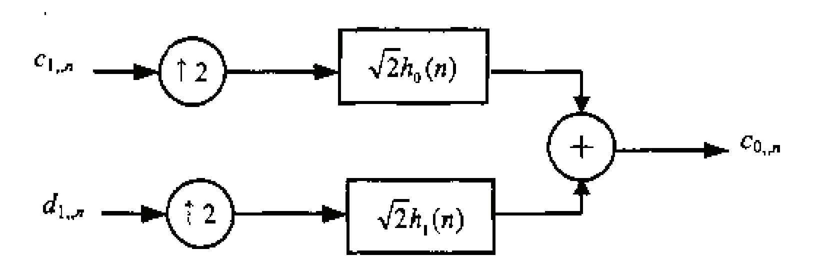
由 (4.11-27) 式可知上式中的积分等于 $2h_0(n-2k)$。因此，

$$\left\langle f_v^1,\phi_{0,n}\right\rangle=\sqrt{2}\sum_{k}c_{1,k}h_0(n-2k) \tag{4.11-31}$$

同理可证

$$\left\langle f_w^1,\phi_{0,n}\right\rangle=\sqrt{2}\sum_{k}d_{1,k}h_1(n-2k) \tag{4.11-32}$$

结合 (4.11-29)、(4.11-31) 及 (4.11-32) 式可得如图 4.11-7 的信号重建方式。



注：圆形符号代表扩增采样，即每两点中添一个零进去。

图 4.11-7 第一层多解析信号的重建

4.11-4 二维离散小波变换

由于静止图像为二维的信号，故有必要将前一节的一维状况推广至二维。最常采用的方式是二维可分离小波变换，亦即将二维小波变换以二个一维的小波变换来实现。相关的小波母函数变成

$$
\begin{aligned}
\phi(x,y) &= \phi(x)\phi(y) \\
\psi^1(x,y) &= \phi(x)\psi(y) \\
\psi^2(x,y) &= \psi(x)\phi(y) \\
\psi^3(x,y) &= \psi(x)\psi(y)
\end{aligned}
\tag{4.11-33}
$$

其中 $\phi(x)$ 与 $\psi(x)$ 分别为一维的尺度与小波函数。此时，

$$
\left\{\psi^l_{mnk}(x,y)\right\} = \left\{2^{-\frac{m}{2}}\psi^l(2^{-m}x-n, 2^{-m}y-k)\right\}, \quad l=1,2,3 \tag{4.11-34}
$$

为规范正交基底。

同上一节，实现离散小波变换比较简单又有效率的方式是通过相对应的滤波器来达成。将原先对一维信号所用的滤波器分别对用在二维信号的行与列，可达二维离散小波变换的效果。整个第一层分解与重建过程分别如图 4.11-8 与 4.11-9 所示。注意到行与列交错实施的部分，否则图像无法重建成功。

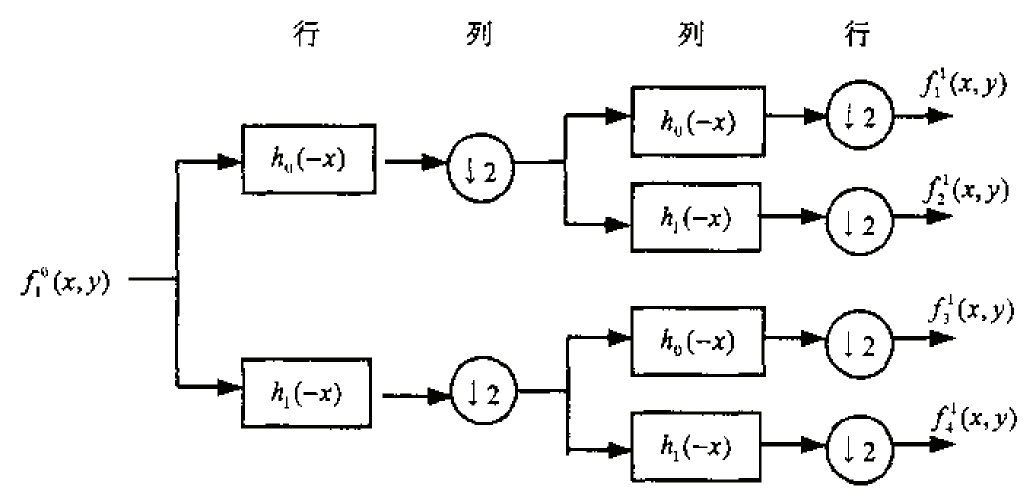


图 4.11-8 离散小波变换图像的分解步骤

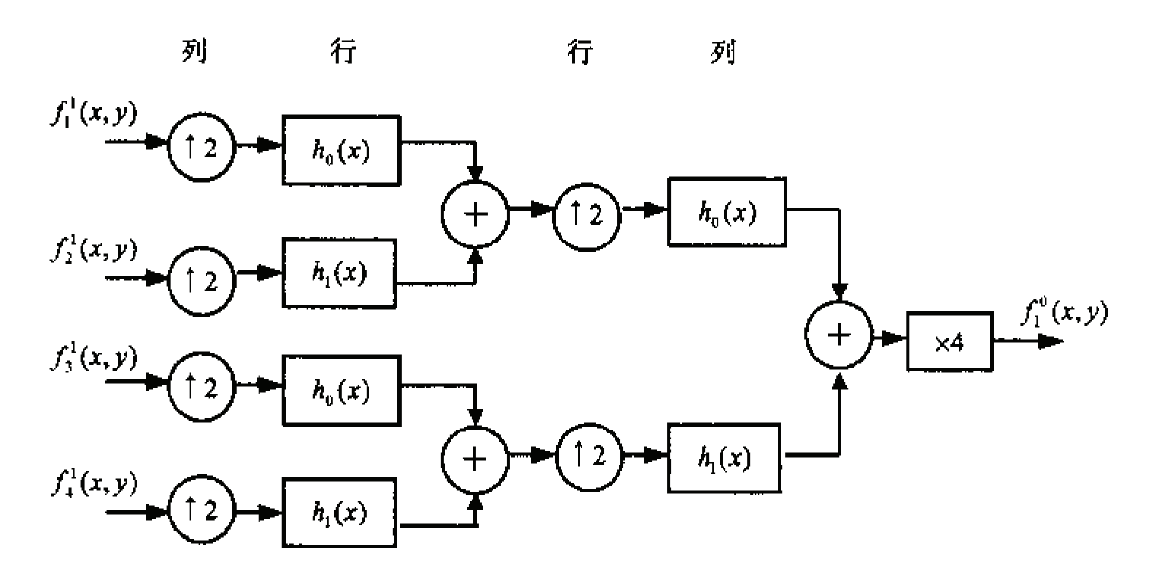
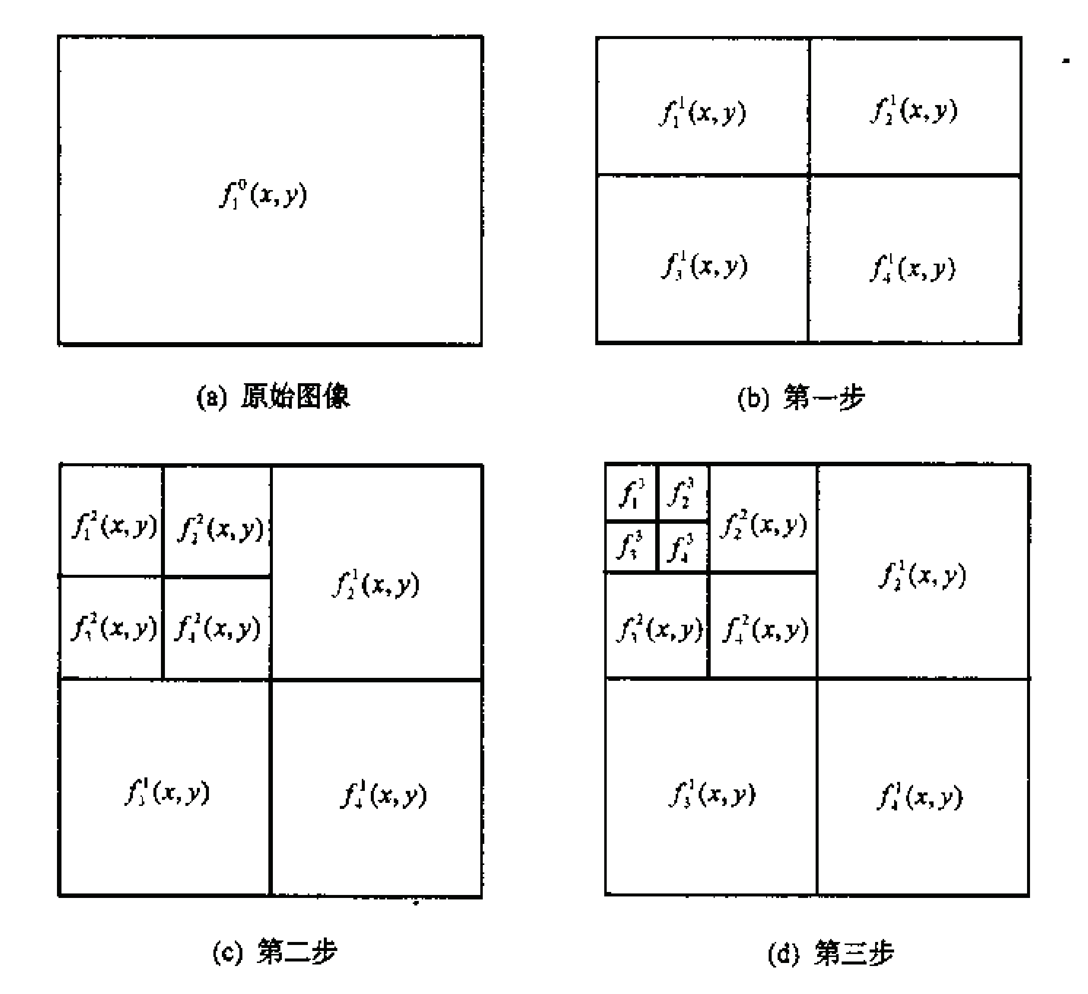


图 4.11-9　离散小波变换图像的重建步骤

图像经过图 4.11-8 所示的分解后得到四个子图像 $f_j^1(x,y)$，$j=1,2,3,4$，每个子图像都可再继续做第二层的分解，因而得到如图 4.11-10 所示的分解结果。



(a) 原始图像　(b) 第一步

(c) 第二步　(d) 第三步

图 4.11-10　二维的离散小波变换

4.11-5 双正交小波变换

具有紧支撑 (compact support) 的规范正交小波变换函数缺乏对称特性。我们希望$\psi(t)$为偶或奇函数。根据研究发现，如果采用二组小波基底$\psi(t)$与$\tilde{\psi}(t)$，前者用来做分解 (或分析)，后者用来做重建 (或合成)，则我们可有具紧支撑的对称小波。两组小波互为对偶 (dual)，且其小波族群$\{\psi_{mn}(t)\}$与$\{\tilde{\psi}_{mn}(t)\}$为双正交 (biorthogonal)，亦即

$$\langle \psi_{mn}, \tilde{\psi}_{jk} \rangle = \delta_{m-j}\delta_{n-k} \tag{4.11-35}$$

双正交变换的分解可写成

$$c_{mn} = \langle f(t), \tilde{\psi}_{mn}(t) \rangle \text{ 或 } d_{mn} = \langle f(t), \psi_{mn}(t) \rangle \tag{4.11-36}$$

而重建可写成

$$f(t) = \sum_m \sum_n c_{mn} \psi_{mn}(t) \text{ 或 } f(t) = \sum_m \sum_n d_{mn} \tilde{\psi}_{mn}(t) \tag{4.11-37}$$

两个小波都可做重建或分解，但其中有一个用在分解时，另一个一定要用在重建。

双正交变换也有相对应的滤波器来实现。例如有人选择 B-Spline 函数 (例如三角函数) 为$\phi(t)$而发展出 cos (z) 的多项式函数的$H_0(z)$，其中$H_0(z)$是滤波器脉冲响应$h_0(n)$的 z 变换。

4.11-6 小波包

在多解析度分析当中，每次输入信号的频谱都被分成低或高的频带，其中高频带直接变成输出，低频带则继续再分解成两个子频带，依此类推。如果每次频带分解不一定只能有两个频带，而且不一定只有低频带被分解，高频带也可继续分解，则由此观念所获得的输出结果就称为小波包 (wavelet packet)。

从数据结构的眼光来看，原始多解析度的分析对应到一个极不平衡的二元树 (binary tree)，小波包则可以是平衡且完满 (complete) 的二元树。图 4.11-11 的 (a) 与 (b) 分别显示这两种树状结构。

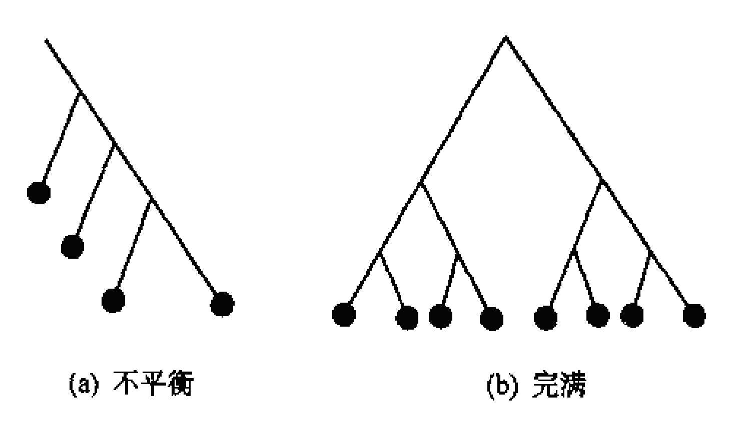


图 4.11-11 二元树

小波包的好处在于通过任意频带的分割达到任意频率解析度的效果。通常伴随小波包的是一个最佳基底选择 (best bases selection) 的自适性算法，以便对某一个特定信号找出若干小波包的组合使其有最适当的解析度。换言之则是保留 (或去掉) 如图 4.11-11 (b) 的完满二元树的一部分，结果变成不完满的树状结构。常采用的选取方式则是以熵 (entropy) 为频带分割时的准则。

以下列举几个熵的准则。在下列叙述中f为信号，f_i则表示f在一规范正交基底的系数。任何熵的表示式 E 都必须是一个相加性的成本函数 (additive cost function) 使得 $E(0)=0$ 且 $E(f)=\sum_i E(f_i)$。

(1) Shannon 熵

$$E(f_i)=-f_i^2\log(f_i^2)\Rightarrow E(f)=-\sum_i f_i^2\log(f_i^2) \tag{4.11-38}$$

并定义 $0\log(0)=0$。

(2) 范数乘幂熵 l^p， $1\leqslant p\leqslant 2$

$$E(f_i)=|f_i|^p\Rightarrow E(f)=\sum_i |f_i|^p \tag{4.11-39}$$

(3) 能量对数熵

$$E(f_i)=\log(f_i^2)\Rightarrow E(f)=\sum_i \log(f_i^2) \tag{4.11-40}$$

并定义为 $\log(0)=0$。

(4) 临界熵

$$E(f_i)=\begin{cases}1, |f_i|>\varepsilon\\ 0, |f_i|\leqslant\varepsilon\end{cases}\Rightarrow E(f)=\#\{|f_i|>\varepsilon\} \tag{4.11-41}$$

亦即信号大于某个临界值ε的次数。

上述熵函数在 Matlab 的小波工具箱 (wavelet toolbox) 中都有现成的函数库可供使用。事实上小波工具箱有使用小波变换所需的各种软件函数库，包括各种小波变换的滤波器系数的产生、多维小波变换的实现、小波变换结果的显示以及许多小波在信号处理上的应用实例等，颇有参考价值。

习 题

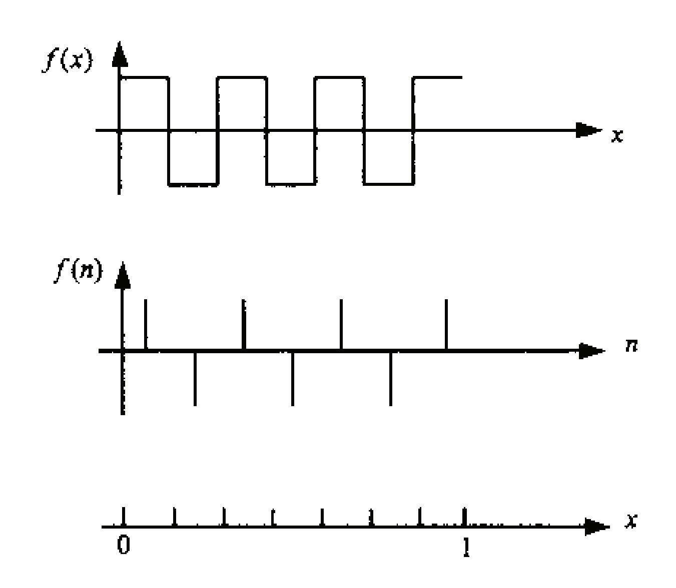
1. P，Q 为已知，试证明当序列系数 $F(k,l)$ 如 (4.1-1) 式所示时，(4.1-6) 式中的误差 σ_ε^2 为最小。
2. 续上题，证明基底图像必形成一 $P=Q=N$ 且 σ_ε^2 为零的完全集。

3. 给一个 2×2 变换矩阵 $\boldsymbol{A}$ 及图像 $\boldsymbol{f}$：

$$\boldsymbol{A}=\frac{1}{2}\begin{bmatrix}\sqrt{3} & 1\\ -1 & \sqrt{3}\end{bmatrix},\qquad \boldsymbol{f}=\begin{bmatrix}2 & 3\\ 1 & 2\end{bmatrix}$$

试计算变换后图像 $\boldsymbol{F}$ 及基底图像。

4. 证明两个函数卷积的傅立叶变换是它们的傅立叶变换的乘积。为简单起见，假设所讨论的为单一变量函数。
5. 试推导并绘出 8 点 FFT 之 DIT 信号流程图。
6. 试推导 (4.3-30) 与 (4.3-31) 式。
7. 试求下列两函数，即 $f(x)$ 与 $f(n)$，在 $[0,1)$ 上的列率。



8. 试求 $N=4$ 时的 Haar 变换矩阵。
9. 试求 $N=8$ 时的斜变换矩阵。
10. 对下列矩阵进行 SVD 变换：

$$\boldsymbol{A}=\begin{bmatrix}0 & 1 & 2 & 1 & 0\\ 1 & 3 & 4 & 3 & 1\\ 2 & 4 & 5 & 4 & 2\\ 1 & 3 & 4 & 3 & 1\\ 0 & 1 & 2 & 1 & 0\end{bmatrix}$$

11. 试选一幅大小为 128×128 的 8 位灰度图像做 DFT 变换、DCT 变换、一层小波变换。其小波变换之滤波器系数分别为：

高通分解系数

[-0.2304 0.7148 -0.6309 -0.0280 0.1870 0.0308 -0.0329 -0.0106]

低通分解系数

[- 0.010 6　0.032 9　0.030 8　- 0.187 0　- 0.028 0　0.630 9　0.714 8　0.230 4]

高通重建系数

[- 0.010 6　- 0.032 9　0.030 8　0.187 0　- 0.028 0　- 0.630 9　0.714 8　- 0.230 4]

低通重建系数

[0.230 4　0.714 8　0.630 9　- 0.028 0　- 0.187 0　0.030 8　0.032 9　- 0.010 6]

(1) 分别显示其变换系数大小的图，并打印之。

(2) 分别显示其变换系数的直方图，并打印之。

(3) 分别显示其反变换（还原）图像，并打印之。

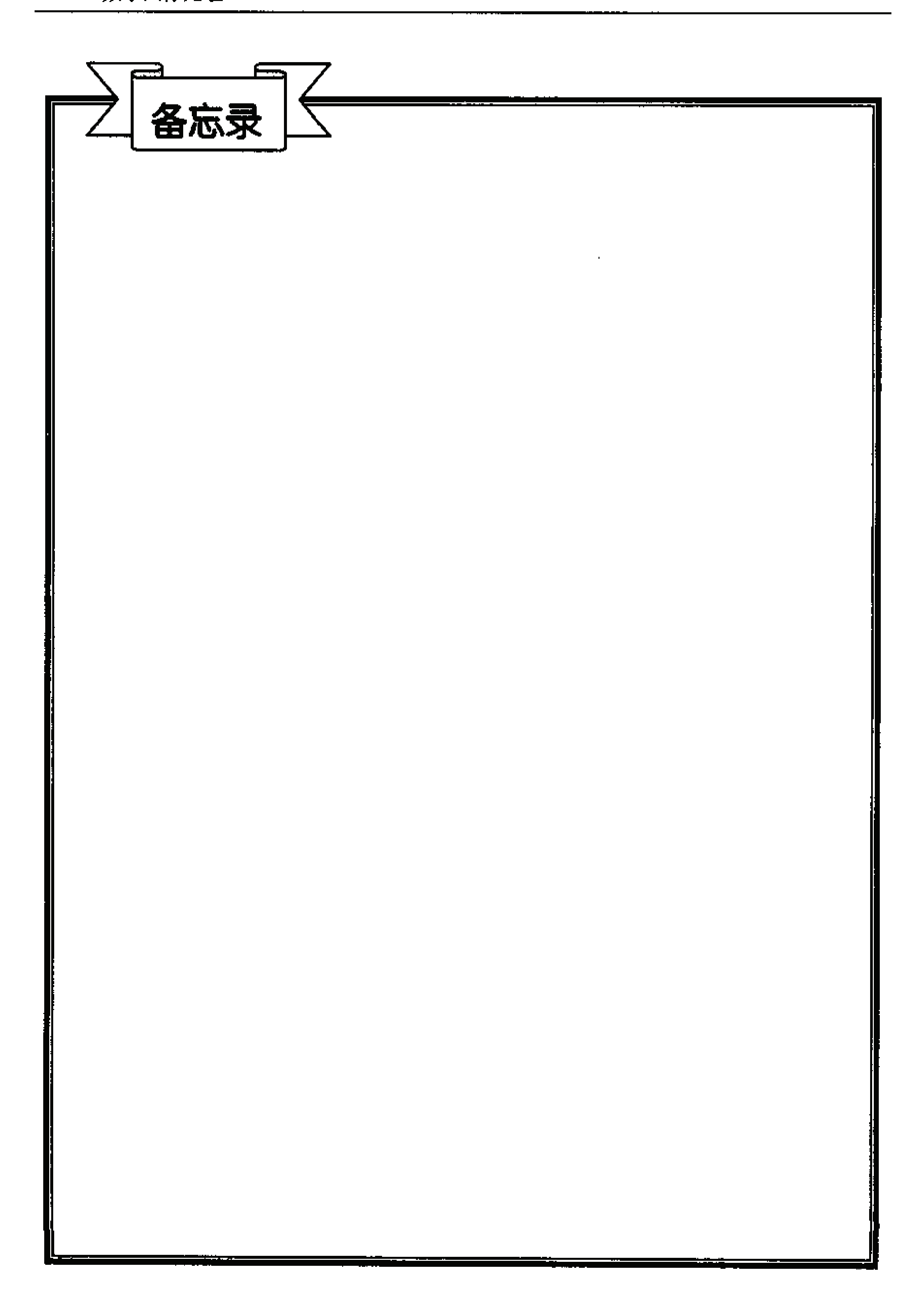
(4) 假设 DFT、DCT 和小波变换系数由低频往高频排列，保留前 25% 低频部分，其余补零，将图像还原，分别显示其图形，并打印之。

(5) 计算 (4) 中还原图像与原图像的 PSNR，其中 PSNR 的定义为：

峰值信噪比：$\text{PSNR} = 20\log_{10}\dfrac{255}{\text{RMSE}}$

均方误差：$\text{RMSE} = \sqrt{\dfrac{1}{MN}\sum_{m=1}^{M}\sum_{n=1}^{N}\left(f(m,n) - \hat{f}(m,n)\right)^2}$

其中 $f(m,n)$ 为原始图像，$\hat{f}(m,n)$ 为重建图像。


备忘录

第五章

图像增强



图像增强技术的主要目的是为了增加图像的视觉效果，让人眼或机器易于辨识。也就是说：处理一幅图像，使其结果对特定的应用来说比原始图像更合用。因此，图像增强技术对已很理想的图像而言是不太有效果，但对一些失真的图形则有明显的成效。此外，图像增强可以防止图像中所代表的重要图像信息的遗漏。例如，一个图像增强系统可以利用高通滤波器来强化图像中物体的边线，使图中的物体更加明显。就像其他图像处理的方法一样，图像增强技术没有所谓必然最佳的方法，而是有各种方法。其中的取舍全视应用的需求，以及图像本身的特性。有的图像增强技术效果很好，但其结构却非常的复杂，以致于硬件难以实现。有的则效果略差一点，但结构简单。本章将介绍多种图像增强的方法。

5.1 点处理增强

所谓点处理 (point processing) 系指针对单一个像素强度的改变而言，这是最简单的一种图像增强技术。以下均假设输入图像点像素的强度为 f，经点处理 T 后的输出强度为 g，即 $g = T[f]$。

5.1-1 简单的灰度变换

最常见的图像缺陷是对比度不足，其原因包括照明强度弱，图像感应器的动态范围不足，甚至可能在图像撷取时光圈设定不当等。而此缺陷可以经由对每点的像素 $f(x,y)$ 作振幅重新调整比例 (amplitude rescaling) 加以改善。图5.1-1 (a) 是一个连续且低对比度图像的典型振幅变换曲线，很容易看出增强后比原始图像的对比度提高了许多。但是这个方法较难实现。相对的图5.1-1(b) 量化振幅 (quantized amplitude) 的转移函数就简单很多，但其代价是有少许的量化误差。

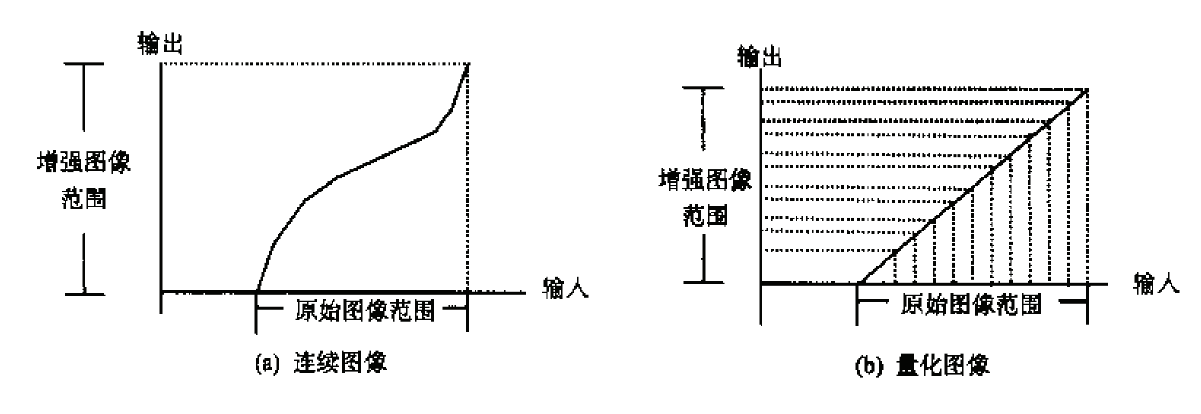
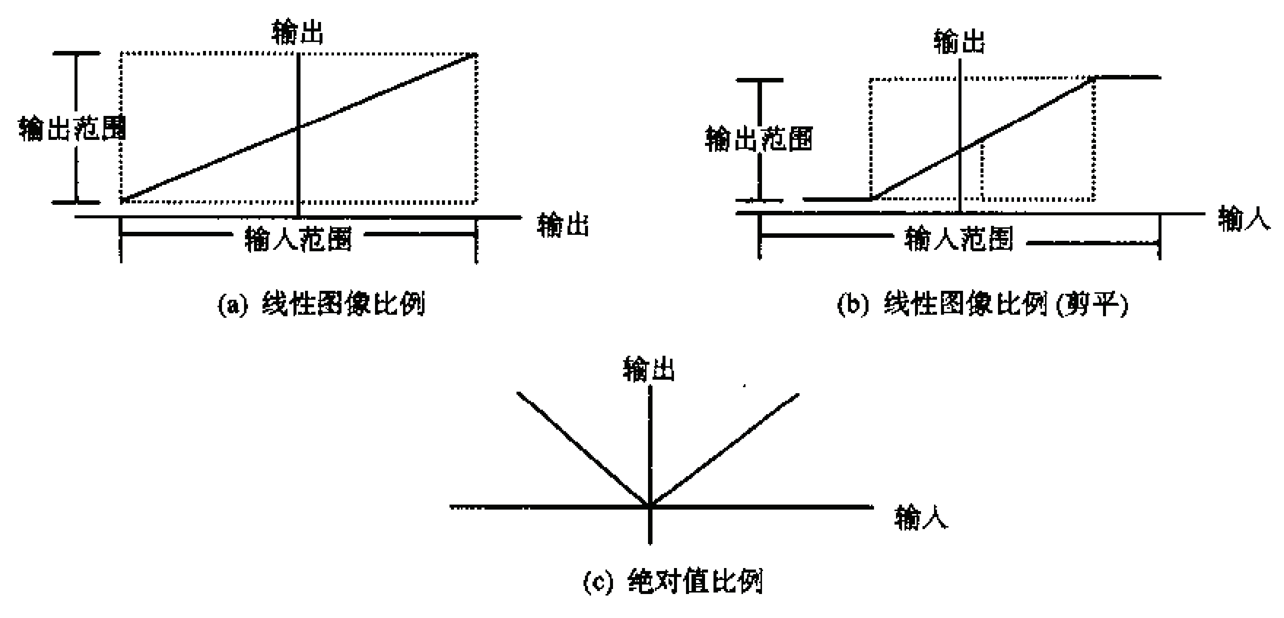


图 5.1-1 图像对比度增强

● **振幅调整**

在处理数字图像时，有时候可能发生处理后的图像与原始图像的数值范围 (range) 不同的情形，甚至在数学运算上产生含有负数的矩阵。显然，图像亮度函数 (intensity function) 必须为正数，此时无法将矩阵直接映射 (mapping) 至图像函数的范围上。图5.1-2显示以重新调整振幅来解决此问题的一些方式。



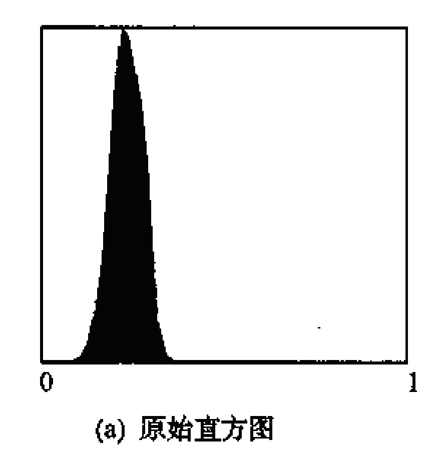
(a) 线性图像比例　　(b) 线性图像比例 (剪平)

(c) 绝对值比例

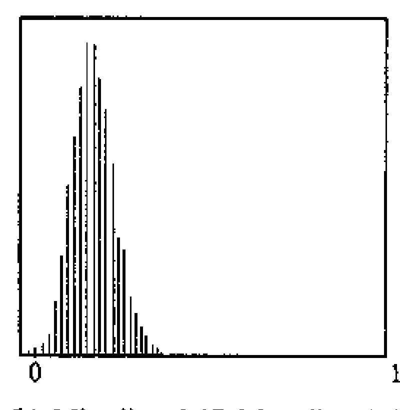
图 5.1-2　图像振幅调整的例子

第一种方法采取线性变换，并将最负的点令为零，于是输出后就没有小于零的点。然后可以将输出在 0 到 1 之间做归一化。第二种方法类似第一种，但是它设了两个极限，输入小于极小值 a 与输入超过极大值 b 以后都剪平(clip)，规一化后分别是极小值 0 与极大值 1，而中间部分仍是线性变换。第三种方法则是将输入值取绝对值，如此一来也能避免负数的产生。一般来说，最受欢迎的是第二种方法，它的效果会比第一种好很多，尤其是当图像中的像素都介于线性 a 至 b 的区段内时。之前已经提过，对比度不足是图像最常见的问题，而对比度不足表示图像的直方图 (histogram) 集中在小范围内，正符合第二种方法的特性。若此时采用第一或是第三种方法只能将负的值变成正值，对于图像的清晰度并没有帮助，这应是第二种方法受欢迎的原因。

下面举一个第二种方法的实例，直接以图像的直方图来比较。我们知道图像对比度越高，其图像函数的直方图分布越广。图 5.1-3(a)是原始图像的直方图，其分布约在 0.20～0.40 之间，其余的地方亮度都为零。这是对比度很差的一幅图像。图 5.1-3 (b) 则取最小值 0.17、最大值为 0.64 做处理，并重新调整比例至 0 与 1 之间，但是效果仍然不显著，因为原始图像 0.40

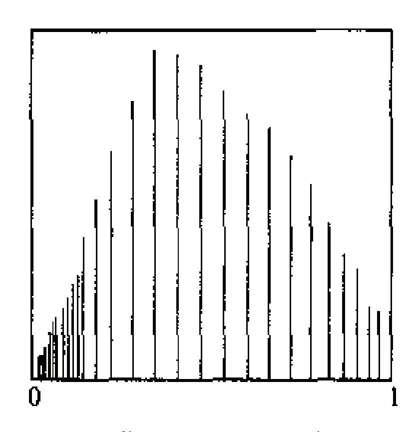


(a) 原始直方图



(b) Min.clip = 0.17, Max.clip = 0.64

图 5.1-3　线性图像调整 (剪平) 的实例 (待续)



(c) Min.clip = 0.24, Max.clip = 0.35

图 5.1-3 (续)

至 0.64 之间亮度全是零，在重新调整比例后亮度为零的区域仍然很大，也就是说对比度没有提高多少。可以发现 (c) 的效果最好，它取最小值为 0.24，最大值为 0.35，这相当于上下切去百分之五的比例，因此所有的点都在线性范围内。至于切除的百分之五，人眼根本观察不出来。这个例子说明，对比度不足的图像，用带剪平的线性图像比例重新调整可以非常有效。

● **对比度修正** (Contrast Modification)

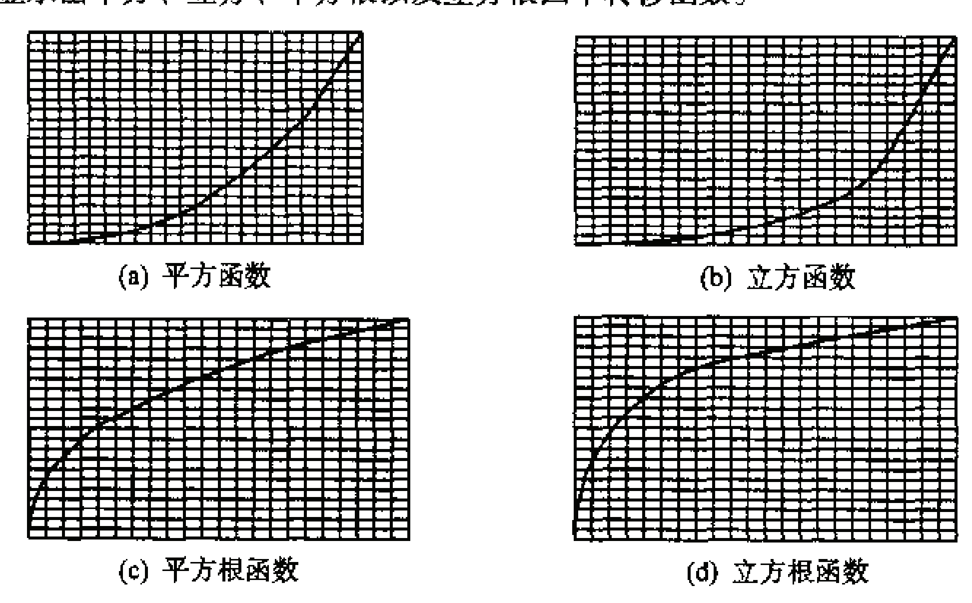
上述振幅调整只适用于对比度集中在一小范围的图像，并不适合动态范围大的图像，此处所需讨论的对比修正则专门处理对比度范围宽广的图像。接下来将介绍几种方法并比较其优缺点及适用场合。

(1) *幂次律点变换* (Power Law Point Transformations)

这种方法的数学定义为

$$g = [f]^p \tag{5.1-1}$$

其中 f 指原始图像的图像函数，p 是次方法则的变量。通常来说，平方 ($p = 2$) 效果最好。图 5.1-4 分别显示出平方、立方、平方根以及立方根四个转移函数。



(a) 平方函数　(b) 立方函数

(c) 平方根函数　(d) 立方根函数

图 5.1-4 次方函数实例

(2) 分段线性函数 (Rubber Band Function)

次方函数是常用也是效果较好的方式，但是我们希望能有一种线性且容易实现的方法以降低计算成本，可是效果又不能太差。如图 5.1-5 所示的分段线性函数则可以满足以上的要求。

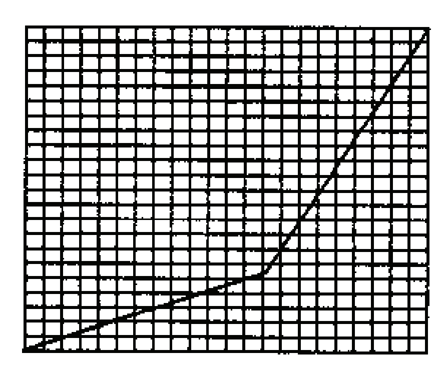


图 5.1-5　分段线性函数

(3) 高斯误差函数 (Gaussian Error Function)

图 5.1-6 表示高斯误差函数，在 f 小的时候 (图形前端) 它的表现像平方函数，在 f 大的时候 (图形后端) 它的表现像平方根函数。其数学定义为

$$g=\frac{\operatorname{erf}\left(\frac{f-0.5}{\sqrt{2}\sigma}\right)+\frac{0.5}{\sqrt{2}\sigma}}{\operatorname{erf}\left(\frac{0.5}{\sqrt{2}\sigma}\right)} \tag{5.1-2}$$

其中

$$\operatorname{erf}(x)=\frac{2}{\sqrt{\pi}}\int_0^x \exp\left(-y^2\right)\mathrm{d}\,y \tag{5.1-3}$$

σ 则为高斯分布的标准差。

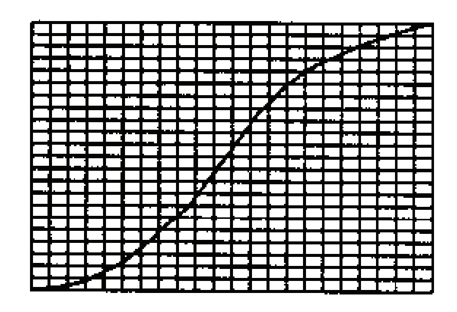


图 5.1-6　高斯误差函数

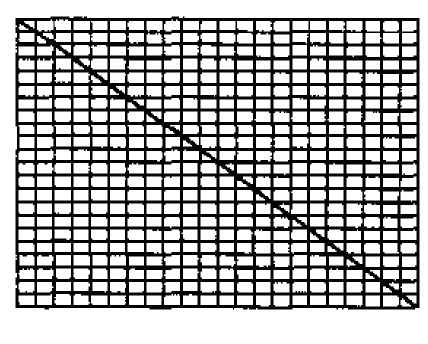
(4) 倒函数 (Reverse Function) 与反函数 (Inverse Function)

倒函数或反函数都可以将明暗度反转，也就是原本最暗的地方变换后变成最亮。其数学定义如 (5.1-4) 与 (5.1-5) 式所示，其图形则如图 5.1-7 所示。很容易看出倒函数是线性的，而

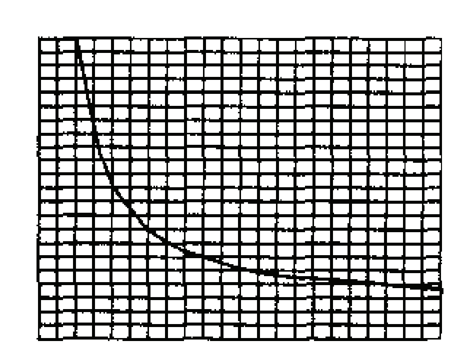
反函数则是非线性。这种亮暗交换的功能形成图像底片的效果，可用在医学图像的显示与幻灯片的制作等。

$$\text{倒函数：}\quad g = 1.0 - f \tag{5.1-4}$$

$$\text{反函数：}\quad g = \begin{cases} 1.0, & 0.0 \leqslant f < 0.1 \\ \dfrac{0.1}{f}, & 0.1 \leqslant f < 1.0 \end{cases} \tag{5.1-5}$$



(a) 倒函数



(b) 反函数

图 5.1-7　亮暗函数

● **动态范围压缩**

有时候经处理后的图像的动态范围大到只能显示出几个像素，例如对图像取傅立叶变换后的系数。通常我们采用以下的强度变换函数

$$g = c\log_{10}(1+|f|) \tag{5.1-6}$$

其中 c 为一常数。假设傅立叶频谱的范围为$[0,\ 5\times10^6]$，则 $\log_{10}(1+|f|)$的范围为 0 到约 6.7。假设我们有一个 8 位的显示系统，即显示范围为 0 到 255，因此常数 c 的选择是 255/6.7≈38。

● **灰度切割** (Gray-Level Slicing)

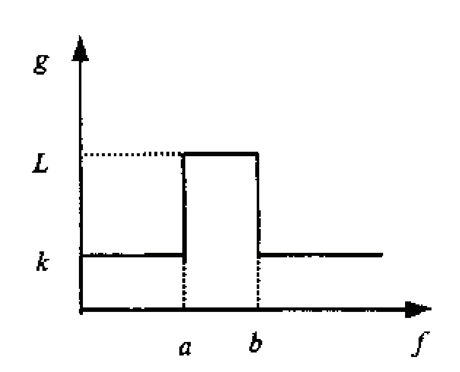
此方法的目的是只强调某范围的灰度，可用来加强图像中某些特定物体的特征，例如将卫星照片中的海水部分或云的部分凸显出来。所采用的基本变换函数为

$$g = \begin{cases} L, & a \leqslant f \leqslant b \\ k, & \text{其他情况} \end{cases} \tag{5.1-7}$$

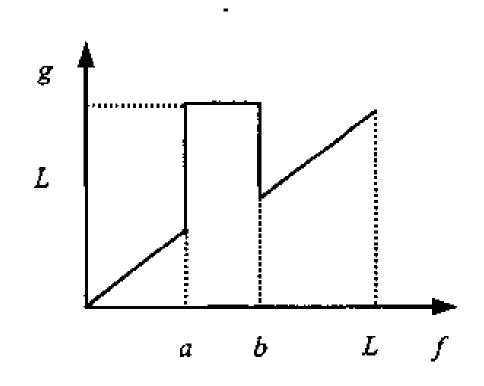
或

$$g = \begin{cases} L, & a \leqslant f \leqslant b \\ f, & \text{其他情况} \end{cases} \tag{5.1-8}$$

此变换函数的曲线分别如图 5.1-8 的 (a) 与 (b) 所示。



(a) 加强在[a,b] 内强度的像素，并把其他像素的强度定为较低的常数值



(b) 加强在[a,b]内强度的像素，其他像素不变

图 5.1-8　灰度切割

● **位平面切割 (Bit-Plane Slicing)**

假设图像的每一个像素都均匀量化成 B 位，则从最低次幂位 (LSB) 到最高次乘幂位 (MSB)，所切割的位平面可想成如图 5.1-9 所示的情况。

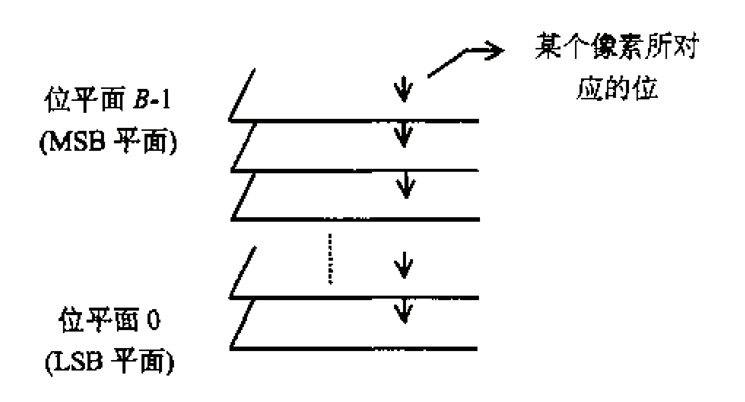


图 5.1-9　一个 B 位图像的位平面表示

令一 B 位图像表示成

$$f = \sum_{i=1}^{B} k_i 2^{B-i} \tag{5.1-9}$$

则从 MSB 平面起算的第 n 个位平面可由下式获得

$$g = \begin{cases} L, & 若 k_n = 1 \\ 0, & 其他情况 \end{cases} \tag{5.1-10}$$

5.1-2　直方图修整 (Histogram Modification)

一个图像的直方图 $p(f)$ 代表灰度为 f 的像素的个数除以总像素所形成的函数，可视为概率密度函数的一个估测。直方图处理的目的是将原图像的直方图修整成某个所要的形式。

例如可将具有狭窄直方图的低对比度图像延展其直方图变成对比度较高的图像而达到图像增强的目的。

● **直方图均衡化**

直方图均衡化的目的是希望获得一个均匀分布直方图的输出图像。设变量 f 代表像素灰度大小在 $[f_{\min}, f_{\max}]$ 区间的一个变量。考虑如下的变换

$$g = T(f) \tag{5.1-11}$$

此变换必须满足下列两个条件：

(1) $T(f)$ 在 $f_{\min} \leqq f \leqq f_{\max}$ 的区间上是单调递增的。

(2) 对于 $f_{\min} \leqq f \leqq f_{\max}$, $g_{\min} \leqq T(f) \leqq g_{\max}$。

将 f 与 g 视为随机变量，其概率密度函数分别为 $p_f(f)$ 与 $p_g(g)$。由基本的概率理论可知

$$\int_{g_{\min}}^{g} p_g(\zeta)\mathrm{d}\zeta = \int_{f_{\min}}^{f} p_f(\zeta)\mathrm{d}\zeta \tag{5.1-12}$$

即两边的累积分布函数相等，记为 $P_g(g) = P_f(f)$。考虑如下变换

$$T(f) = P_f(f) = \int_{f_{\min}}^{f} p_f(\zeta)\mathrm{d}\zeta \tag{5.1-13}$$

显然此函数满足上述 (1) 和 (2) 的条件。

假设输出是概率密度函数如下的均匀分布

$$p_g(g) = \frac{1}{g_{\max} - g_{\min}}, \quad g_{\min} \leqslant g \leqslant g_{\max} \tag{5.1-14}$$

则由 (5.1-12) 到 (5.1-14) 式可得如下变换函数

$$g = (g_{\max} - g_{\min})P_f(f) + g_{\min} \tag{5.1-15}$$

将上述观念延伸到灰度在 $[f_{\min}, f_{\max}]$ 的真实图像来。f 现在是有限个离散值，原先概率密度函数变成下列概率

$$p_f(f) = \frac{n_f}{n}, \quad f_{\min} \leqslant f \leqslant f_{\max} \tag{5.1-16}$$

其中 n_f 表示灰度 f 出现的像素个数，n 则为所有像素的个数。因此 (5.1-13) 式的离散形式为

$$T(f) = P_f(f) = \sum_{i=f_{\min}}^{f} \frac{n_i}{n}, \quad f_{\min} \leqslant f \leqslant f_{\max} \tag{5.1-17}$$

将此结果代入 (5.1-15) 式即得均衡化变换后的输出为

$$g=\left(g_{max}-g_{min}\right)\sum_{i=f_{min}}^{f}\frac{n_i}{n}+g_{min}, \quad f_{min}\leqslant f\leqslant f_{max} \tag{5.1-18}$$

● **直方图修整**

在直方图均衡化中，我们希望输出的概率密度函数为

$$p_g(g)=\frac{1}{g_{max}-g_{min}}, \quad g_{min}\leqslant g\leqslant g_{max}$$

我们也可以得到直方图依其他的输出所需的概率密度修整，如表 5.1-1 所示。

表 5.1-1 直方图的输出概率密度及其相对应的变换函数

输出概率密度模型	变换函数
均匀型 (Uniform)	
$p_g(g)=\dfrac{1}{g_{max}-g_{min}}$	$g=(g_{max}-g_{min})P_f(f)+g_{min}$
指数型 (Exponential)	
$p_g(g)=\alpha\exp[-\alpha(g-g_{min})]$	$g=g_{min}-\dfrac{1}{\alpha}\ln[1-P_f(f)]$
雷利型 (Rayleigh)	
$p_g(g)=\dfrac{g-g_{min}}{\alpha^2}\exp\left[-\dfrac{(g-g_{min})^2}{2\alpha^2}\right]$	$g=g_{min}+\left[2\alpha^2\ln\left(\dfrac{1}{1-P_f(f)}\right)\right]^{\frac{1}{2}}$
立方根型 (Cube Root)	
$p_g(g)=\dfrac{1}{3}\dfrac{g^{-2/3}}{g_{max}^{1/3}-g_{min}^{1/3}}$	$g=\left[\left(g_{max}^{1/3}-g_{min}^{1/3}\right)P_f(f)+g_{min}^{1/3}\right]^3$
对数型 (Logarithmic)	
$p_g(g)=\dfrac{1}{g[\ln(g_{max})-\ln(g_{min})]}$	$g=g_{min}\left[\dfrac{g_{max}}{g_{min}}\right]^{P_f(f)}$

● **指定直方图** (Histogram Specification)

虽然直方图均衡化或直方图修整是很有用的方法，但是因为只是产生具有近似几个概率密度的结果，并不太适合做交互式 (interactive) 图像增强。有时我们需要有特定形式的直方图，以便突显某段灰度值的范围。

同 (5.1-13) 式，假设所需的输出概率密度函数为 $p_g(g)$，定义其累积分布函数所构成的变换如下

$$R(g)=P_g(g)=\int_{g_{min}}^{g}p_g(\zeta)\mathrm{d}\zeta \tag{5.1-19}$$

由 (5.1-12)、(5.1-13) 与 (5.1-19) 式可得

$$R(g)=T(f) \tag{5.1-20}$$

因而待求的灰度 g 可写成

$$g=R^{-1}[T(f)] \tag{5.1-21}$$

接着讨论上述观念的离散版本。首先定义

$$T(f)=\sum_{i=f_{\min}}^{f}\frac{n_i}{n}, \quad f_{\min}\leqslant f\leqslant f_{\max} \tag{5.1-22}$$

$$R(g)=\sum_{i=g_{\min}}^{g}\frac{n_i}{n}, \quad g_{\min}\leqslant g\leqslant g_{\max} \tag{5.1-23}$$

我们要决定的是 f 到 g 之间的对应。给一 f 值，由 (5.1-22) 式可得 $T(f)$，然后由 (5.1-23) 式找到一最小的 g 值使得 $R(g)\geqslant T(f)$，此 g 值即为 f 的对应值。

● **局部增强**

一个图像的直方图是图像总体的统计特性，依此特性所得的图像增强效果，对整体而言或许有不错的平均表现，但是对图像的某些部分未必最佳，由此衍生出局部增强的概念。

一般局部增强的做法是以待处理像素为中心连同其周边若干像素形成直方图，再采用先前所述的直方图均衡化或指定直方图等的方法。

5.1-3 其他点处理法

● **方差均衡化**

直方图均衡化的目标是希望输出有近似均匀的概率密度函数，而方差均衡化则希望获得均匀的方差。这个做法适合局部增强，可使对比度或方差过低的较均匀部分突显出彼此的差异。考虑一中心为 $f(x,y)$ 的局部图像。设 $f(x,y)$ 所对应的变换点为 $g(x,y)$，则

$$g(x,y)=\frac{kM}{\sigma(x,y)}[f(x,y)-m(x,y)]+m(x,y) \tag{5.1-24}$$

其中 $m(x,y)$ 与 $\sigma(x,y)$ 分别为此局部图像的灰度平均值与偏差量，M 是整幅图像的灰度平均，而 k 是在 0 与 1 之间的调整参数。当偏差量 $\sigma(x,y)$ 小的时候，$f(x,y)$ 与 $m(x,y)$ 的差异量可获得较大的放大倍率，而达到加大对比度的效果。

● **图像相减**

将两图像逐点相减也是一种图像增强的方式。例如，以身体某部分的医学图像为对象，再注入特殊感光化学液体至血管中所得的画面与其相减，将相减过的画面连续播放，就可看

出这些化学液体是如何在血管中流动的。

● **图像平均**

假设一个图像感应器受到平均值为零的高斯噪声的影响，因此得到的是带噪声的图像。有一个消除此噪声的方法就是对同一画面多取几张图像，再取这些图像的平均图像。理论上，由于所受到的是平均值为零的高斯图像，求平均的作用可抵消噪声的效应，而且用的图像愈多效果愈好。

5.2　空间滤波

采用模板 (mask) 逐点对图像做处理的方法称为空间滤波，这是相对于采取如傅立叶等变换所得的频域滤波法而言。

5.2-1　平滑滤波

平滑滤波器本质上是一个低通滤波器，它主要用来使图像模糊或降低噪声。对图像识别的目的而言，图像模糊可去除妨碍重要特征抽取的小细节并使断线相连。对于脉冲型的高频噪声，平滑滤波将可减轻此噪声的效应。

● **平均滤波**

一个 3×3 平均滤波器所采用的模板如图 5.2-1 所示，由此图可明显看出其名称的由来。一般常用的还有 5×5 或 7×7 的模板。愈大的模板模糊效果愈强，相当于此滤波器的截止频率愈来愈低，高频部分被滤去愈多。

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

图 5.2-1　3×3 平均滤波器模板

● **中值滤波器**

平均滤波往往不只是把干扰去除，还常把图像的边缘变模糊，因而造成视觉上的失真。如果目的只是要把干扰去除，而不是刻意让图像模糊，则中值滤波是很好的选择。

首先把模板内所涵盖的像素灰度值由小到大排列，中值是指排序在中间的那一个值，此值即为滤波器的输出。此法特别适合用在有很强的胡椒粉式或脉冲式的干扰时，因为这些灰度值的干扰值与其邻近像素的灰度值有很大的差异，因此经排序后取中值的结果是强迫将此干扰点变成与其邻近的某些像素的灰度值一样，达到去除干扰的效果。由中值滤波器的原理不难看出中值滤波是一个非线性的操作。

另一种非线性的做法是先计算周边像素灰度的平均值，若所考虑的像素的灰度与此平均

值差异量超过一定的临界值，则判定此像素为干扰，而用先前计算所得的平均值来代替。

5.2-2 锐化滤波

锐化 (sharpening) 滤波本质上是一个高通滤波器，它主要用来使图像的细节(detail)或边缘更突显，而达到图像增强的目的。

● **基本高通空间滤波**

一个 3×3 高通空间滤波器所采用的模板如图 5.2-2 所示。若模板中心点对应具有较大灰度的像素，则经此滤波后，此像素与其旁边像素之间的灰度的差异会被放大，反之，此模板对灰度变化相当慢的平滑区域，其输出将非常小。极端状况是若模板所涵盖范围内的灰度都一样时，则不管原来是多大灰度值，其输出恒为零，这表示此种模板有降低整体图像平均值使图像整体变暗的缺点。另外，实际的输出有负值的可能性，必须做大小的调整。

$$\frac{1}{9}\begin{bmatrix}-1 & -1 & -1\\-1 & 8 & -1\\-1 & -1 & -1\end{bmatrix}$$

图 5.2-2 3×3 高通空间滤波器模板

● **高频加强 (High-Frequency-Emphasis)滤波**

此种滤波是先将原始图像乘上一个大于 1 的倍率再减去此图像经低通滤波后的结果。此种滤波器的一个 3×3 模板如图 5.2-3 所示，其中 α 为放大倍率。注意到 $\alpha=1$ 时，此结果与图 5.2-2 中所示的模板完全一样。

$$\frac{1}{9}\begin{bmatrix}-1 & -1 & -1\\-1 & 9\alpha-1 & -1\\-1 & -1 & -1\end{bmatrix}$$

图 5.2-3 3×3 高频加强滤波器的模板

● **差分型滤波器**

差分 (differencing) 型滤波器的主要概念是将图像物体的边缘 (edge) 与其邻近像素间的灰度加以放大，达到凸显物体边缘轮廓的图像增强效果。此种方法亦可作为图像分割中检测物体边缘之用。

最常见的差分型滤波器是利用梯度的方法。对于一个函数 $f(x,y)$，其梯度为

$$\nabla f=\frac{\partial f}{\partial x}\boldsymbol{i}+\frac{\partial f}{\partial y}\boldsymbol{j}=f_x\boldsymbol{i}+f_y\boldsymbol{j} \tag{5.2-1}$$

其大小为

$$|\nabla f| = \left(f_x^2 + f_y^2\right)^{\frac{1}{2}} \tag{5.2-2}$$

对于一个离散型函数 $f(m,n)$，类似 (5.2-2) 式的梯度的大小定义为

$$|\nabla f| = \left(f_m^2 + f_n^2\right)^{\frac{1}{2}} \tag{5.2-3}$$

其中 f_m 代表 m 的方向 (列的方向) 上的梯度分量，f_n 则代表 n 方向 (行的方向) 上的梯度分量。有时为了节省计算量，我们采用

$$|\nabla f| = |f_m| + |f_n| \tag{5.2-4}$$

代替 (5.2-3) 式。

最简单的离散型梯度表示是相邻像素灰度间的差量。例如列方向上的梯度分量为

$$f_m = f(m,n) - f(m+1,n) \tag{5.2-5}$$

而行方向上的梯度分量为

$$f_n = f(m,n) - f(m,n-1) \tag{5.2-6}$$

以 3×3 模板表示，则分别如图 5.2-4 的 (a) 与 (b) 所示。

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 0 \end{bmatrix} \qquad\qquad \begin{bmatrix} 0 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

(a) 列方向　　　　(b) 行方向

图 5.2-4　实现梯度分量的一个 3×3 模板

其他常见的模板如图 5.2-5 所示。

运算模板	列方向	行方向
隔点灰度差	$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$
Roberts	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix}$
Prewitt	$\frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$	$\frac{1}{3}\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$

Sobel	$\frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$	$\frac{1}{4}\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$
Frei-Chen	$\frac{1}{2+\sqrt{2}}\begin{bmatrix} 1 & \sqrt{2} & 1 \\ 0 & 0 & 0 \\ -1 & -\sqrt{2} & -1 \end{bmatrix}$	$\frac{1}{2+\sqrt{2}}\begin{bmatrix} -1 & 0 & 0 \\ -\sqrt{2} & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix}$

图 5.2-5 常见的梯度模板运算

5.3 频域的图像增强法

对一图像进行傅立叶变换后，物体边缘或灰度变动较剧烈的部分反应在傅立叶系数中的“高频部分”(即坐标较大者)，平滑的部分则反应在傅立叶系数中的“低频部分”(即坐标较小者)。依此原理，若要取低通滤波的效果，可将较高频部分的系数衰减其大小，甚至全部为零，再对所有处理过的变换系数取反傅立叶变换即可得到低通滤波的效果。反之，若要有高通滤波的效果，则将较低频系数衰减。

5.3-1 由频域指定滤波器特性

设 $f(m,n)$ 为原图像，$h(m,n)$ 为滤波器的脉冲响应，$g(m,n)$ 为图像滤波后的结果，则

$$g(m,n)=f(m,n)*h(m,n) \tag{5.3-1}$$

其中*代表卷积。对 (5.3-1) 式取傅立叶变换可得

$$G(k,l)=F(k,l)H(k,l) \tag{5.3-2}$$

因此在频域中，我们可依不同 $H(k,l)$ 的选择来达到低或高通滤波的效果。
例如

$$H(k,l)=\begin{cases}1, & D(k,l)\leq D_0 \\ 0, & D(k,l)>D_0\end{cases} \tag{5.3-3}$$

代表一个理想的低通滤波器，其中 D_0 称为截止频率，而

$$D(k,l)=\left(k^2+l^2\right)^{\frac{1}{2}} \tag{5.3-4}$$

代表点 (k,l) 到原点间的欧几里德距离。同理，理想的高滤波器为

$$H(k,l)=\begin{cases}0, & D(k,l)\leq D_0 \\ 1, & D(k,l)>D_0\end{cases} \tag{5.3-5}$$

由连续函数的傅立叶变换对

$$f(t)=\frac{\sin(t)}{\pi t}\Leftrightarrow F(\omega)=\begin{cases}1, & |\omega|<1\\ 0, & |\omega|>1\end{cases}$$

我们可以推测 $H(k, l)$ 的反变换 $h(m, n)$ 必然是一个如 $\sin(t)/t$ 的取样(sampling)函数，由 (5.3-1) 式可推测出所得的结果会有所谓的振铃 (ringing) 效应，因此我们摒弃理想滤波器而采用实际滤波器，例如低通与高通巴特沃斯 (Butterworth) 滤波器分别为

$$H(k,l)=\frac{1}{1+[D(k,l)/D_0]^{2n}} \tag{5.3-6}$$

与

$$H(k,l)=\frac{1}{1+[D_0/D(k,l)]^{2n}} \tag{5.3-7}$$

其中 n 代表阶数，n 愈大滤波器响应变化愈快速；反之则愈慢。

5.3-2　同态(Homomorphic)滤波

假设一个图像 $f(m,n)$ 可由光强度 $i(m,n)$ 与物体反射光强度 $r(m,n)$ 的分量乘积来决定，即

$$f(m,n)=i(m,n)\cdot r(m,n) \tag{5.3-8}$$

将 (5.3-8) 式取对数可得

$$\ln[f(m,n)]=\ln[i(m,n)]+\ln[r(m,n)] \tag{5.3-9}$$

一般而言，光的强度分量 $i(m,n)$ 对所有 (m,n) 而言均为定值或变化很缓慢，而反射光强度分量 $r(m,n)$ 则变化较大，特别是在不同物体间的交接处。因此若对 (5.3-9) 式取傅立叶变换，则低频傅立叶系数反应在 $\ln[i(m,n)]$ 上，高频傅立叶系数则反应在 $\ln[r(m,n)]$ 上。

接下来我们可依照上节的观念，指定所需要的滤波器特性，设为 $H(k,l)$。则由 (5.3-9) 式可得

$$H(k,l)\cdot\mathcal{F}\{\ln[f(m,n)]\}=H(k,l)\cdot\mathcal{F}\{\ln[i(m,n)]\}+H(k,l)\cdot\mathcal{F}\{\ln[r(m,n)]\} \tag{5.3-10}$$

这表示我们所取的滤波器 $H(k,l)$ 可分别改变光强度与反射光强度的特性，因此我们可以做到同时降低图像动态范围 (低频处 $H(k,l)<1$)，又增加对比度 (高频处 $H(k,l)>1$) 的结果。

最后的程序是要将 (5.3-10) 式取反傅立叶变换后再取指数函数，才得到图像增强后的结果 $g(m,n)$。整个流程图显示于图 5.3-1 中。

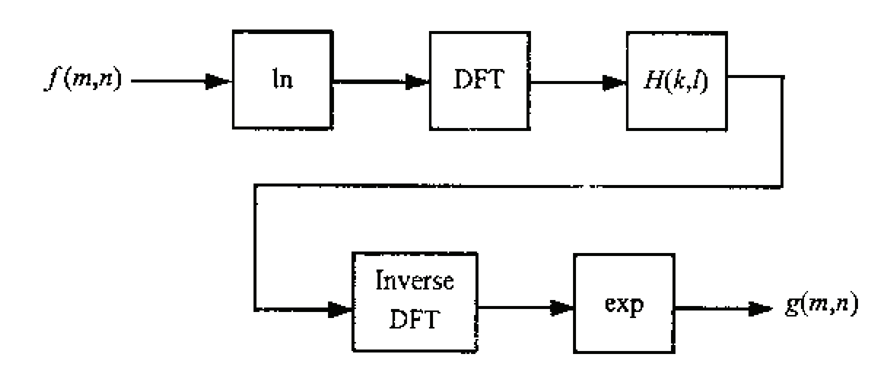


图 5.3-1　同态滤波图像增强法的流程图

5.4　彩色图像增强

将彩色用到图像处理中有两个主要原因：

① 在自动化图像分析中，彩色是一种有效的描述，这种彩色描述常常可简化目标识别以及从背景中提取目标的工作。

② 以人做图像分析时，人眼能分辨几千种彩色色调和强度，却只能分辨二十几个灰度级。

彩色图像处理分为两个主要领域：全彩色 (full color) 和伪彩色 (pseudo-color) 处理。在第一类处理中，待处理的图像是用全彩色感应器，例如彩色摄影机以及彩色扫描仪所获取。在第二类彩色处理中，对于每一个色调给一个特定的单色强度或强度范围。直到最近以前，大多数彩色处理是在伪彩色这个层次上进行。1980 年起彩色处理技术获得很大的发展，因为当时已经制造出彩色感应器，而且用彩色图像处理的硬件价格也趋于合理。因为这些发展的结果，使得全彩色图像处理技术获得了广泛应用。

5.4-1　彩色模型

彩色模型的用途是便于以某种标准来指定彩色。事实上，一种彩色模型是指规定一个三维坐标系统和一个子空间，在此子空间内每种彩色用一个点来表示。现今使用的大多数彩色模型有两种。一种是硬件导向的 (如显示器和打印机)，另一种是应用导向 (如动画彩色图形制作)。实际中常使用且较为简单的是 RGB (红、绿、蓝) 模型，以下简述此模型。

在 RGB 模型中，每种颜色是以它的红、绿、蓝的频谱分量来表现。该模型是建立在直角坐标基础上，如图 5.4-1 所示，其中 RGB 值是在三个顶点上；青色、紫红和黄在另外三个顶点上，黑在原点、白色在点 (1,1,1) 上，而灰色位于黑到白之间的区段上。所有的色彩都在立方体的边缘及内部中，它是由从原点出发的向量所定义的。为了方便起见，我们假设所有彩色的数值已经过归一化，也就是说，所有 RGB 的值都在[0, 1]范围内。

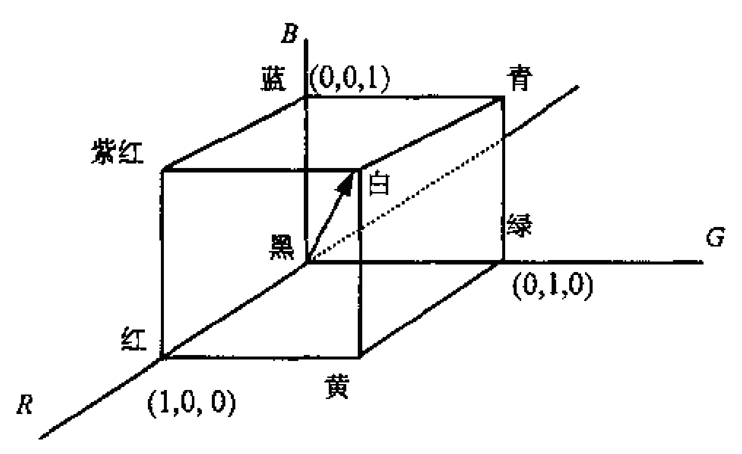


图 5.4-1　RGB 彩色模型

RGB 彩色模型下的图像是由三个独立的图像平面所组成，每个原色对应一个平面。当这三个图像平面传给 RGB 显示器时，这三幅图像组合起来便成为一幅彩色图像。所以当图像本身本来就是用三个彩色平面所表示时，将 RGB 模型用于图像处理是有意义的。另一方面，用于获取数字图像的大多数彩色摄影机使用的是 RGB 方式，所以 RGB 彩色模型在图像处理中是一个重要的模型。

另一种重要的彩色模型是 yuv 模型。图 5.4-2 表示 RGB 彩色模型与 yuv 彩色模型的变换。其中 y 代表亮度 (lumiance)，u 为色调 (hue)，v 则是饱和度 (saturation) (编者按：u、v 应为色差信号 (chrominance))。RGB 彩色模型与 yuv 彩色模型之间的变换关系可以用矩阵转换方式描述，如 (5.4-1) 式所示。至于其他种类的模型，彼此之间也都可以任意变换。

$$\begin{bmatrix} y \\ u \\ v \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.523 & 0.311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{5.4-1}$$

采用 yuv 模型的好处是亮度 y 以及与颜色相关的 u,v 被单独分离开，因此许多以前所说的图像处理法，例如直方图图像增强法等都可以针对 y 来进行而不会影响原图像的颜色。

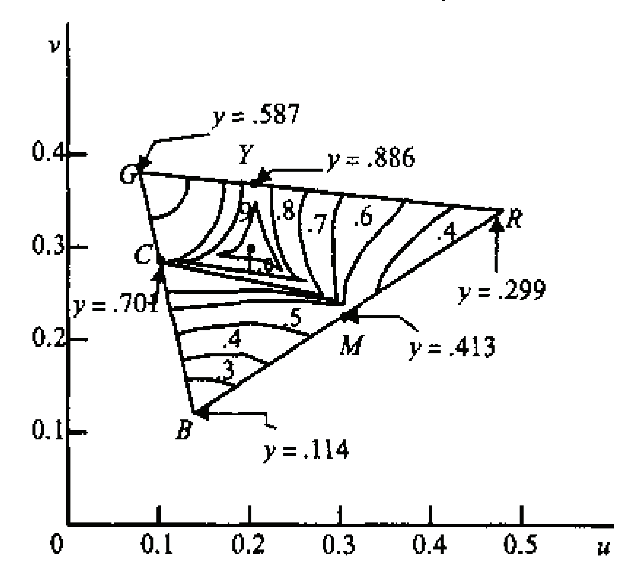


图 5.4-2　RGB 彩色模型与 yuv 彩色模型的变换

5.4-2　伪彩色

伪彩色处理主要的效果是将原本灰阶的图像，经由线性或非线性变换成彩色图像。一个典型的应用就是给古老的黑白图像上颜色使其看来更生动活泼。其数学定义为

$$\begin{aligned} R(x,y) &= Q_R\{f(x,y)\} \\ G(x,y) &= Q_G\{f(x,y)\} \\ B(x,y) &= Q_B\{f(x,y)\} \end{aligned} \tag{5.4-2}$$

其中Q_R、Q_G、Q_B是线性或非线性操作。如图 5.4-3 所示，图中红色变换、绿色变换和蓝色变换分别代表Q_R、Q_G、Q_B。而$f(m,n)$原本是灰阶图像、经过三个独立变换后、形成红、绿、蓝个别分量。然后把这三个结果单独送到彩色显示器的红、绿、蓝三个电子枪上，这样就产生一幅彩色合成图像。

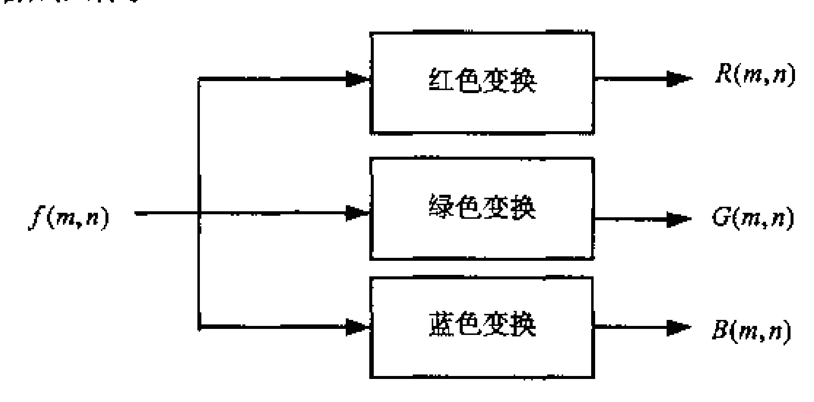


图 5.4-3　伪彩色图像处理的基本方块图

5.4-3　假彩色

假彩色 (false color) 也是一种彩色图像处理的方法，它也是点对点变换。与伪彩色不同的是，假彩色处理的原始图像必须是彩色图像。经过其特殊的矩阵变换，可以将彩色图像中原本的颜色加以变化。其数学定义为

$$\begin{aligned} R_D &= Q_R\{F_1,F_2,\cdots\} \\ G_D &= Q_G\{F_1,F_2,\cdots\} \\ B_D &= Q_B\{F_1,F_2,\cdots\} \end{aligned} \tag{5.4-3}$$

其中$F_i, I=1,2,\cdots$代表不同的颜色。与伪彩色一样，Q_R、Q_G、Q_B是线性或非线性操作。也许这样仍然无法看出其运作效果，下面给一个线性操作的形式：($F_1=R$；$F_2=G$；$F_3=B$)。

$$\begin{bmatrix} R_D \\ G_D \\ B_D \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{5.4-4}$$

若矩阵为

$$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \tag{5.4-5}$$

经过计算可得

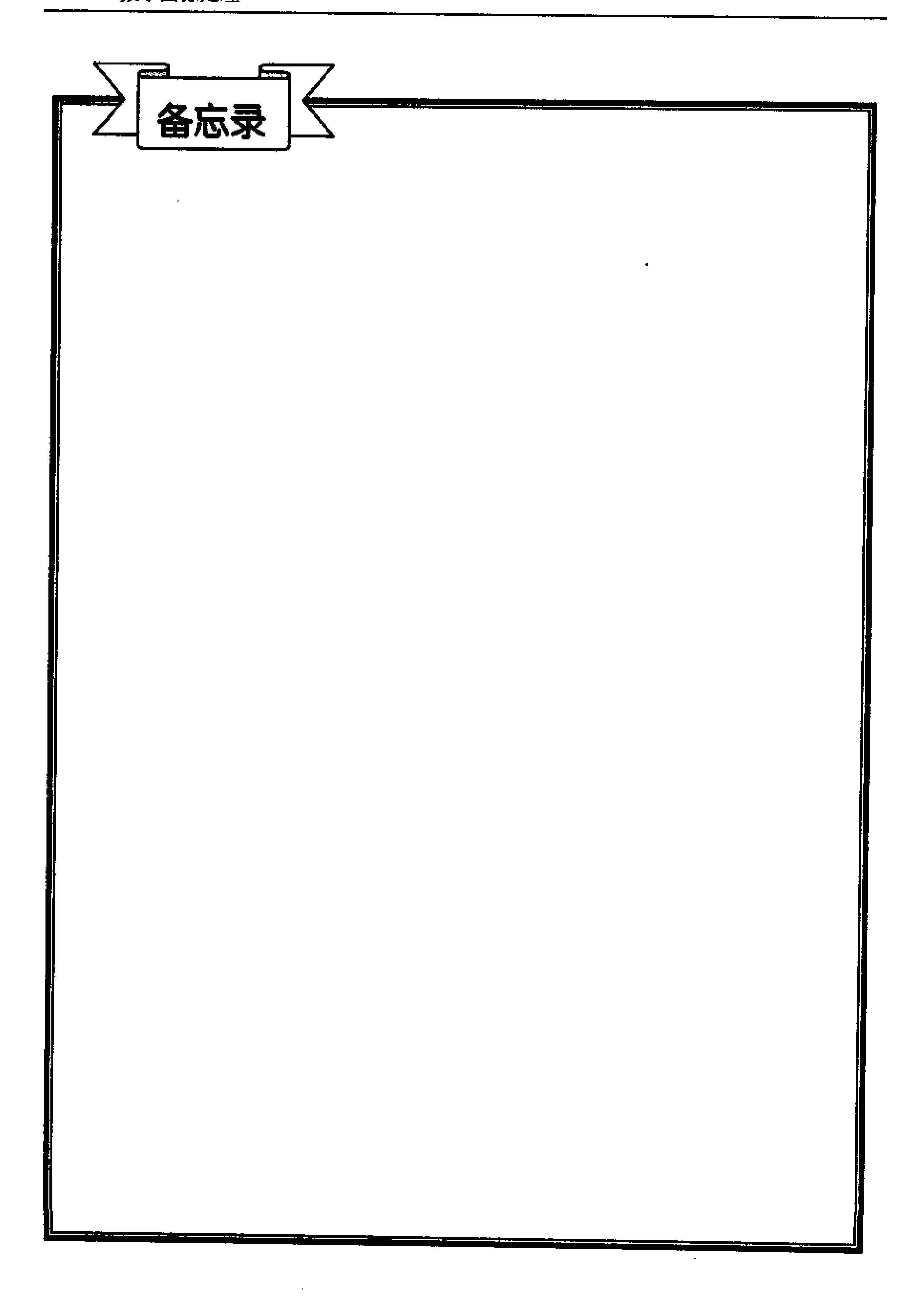
$$\begin{aligned} R_D &= G \\ G_D &= B \\ B_D &= R \end{aligned} \tag{5.4-6}$$

则原本图像的变化如下：绿色变红色；蓝色变成绿色；红色变蓝色。

由前例可知，经由假彩色处理，可以利用适当的变换矩阵改变原始图像中的色彩。这种技术目前已广泛被应用在制造特殊效果上。

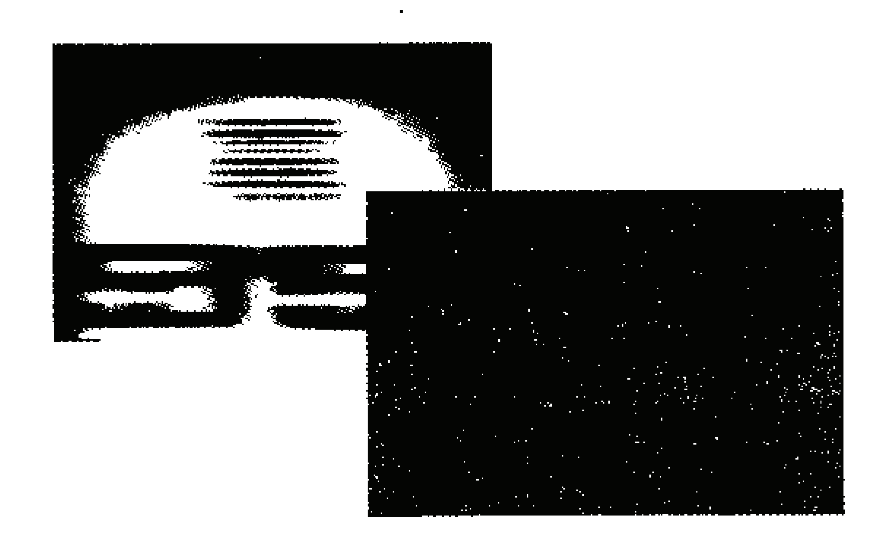
习　题

1. 你认为将图像放大是一种图像增强的处理吗?为什么?
2. 以程序实现下列两种简单的图像放大方法：
   (1) 零阶保持 (zero-order hold)：每个图像先沿行 (列) 方向复制一次，完成后的图像再沿列 (行) 的次序再复制一次。
   (2) 一阶保持 (first-order hold)：先沿行 (列) 的方向，在两像素间插一个像素、其灰度值为前后两像素灰度值的平均，完成后的图像，将上述程序再沿列 (行) 的方向做一次。
   取一 64×64 的图像将其放大至 128×128 以及 256×256，并比较上述两种方法的效果。
3. 一个单调递增的函数是否适合一般图像振幅调整法的输入输出间的关系?为什么?
4. 试取一低对比度图像以对比度修正法改善其对比度。
5. 重复上题改以任一直方图处理法进行。
6. 推导出表 5.1-1 中的变换函数。
7. 以指定直方图法对一低对比度图像进行图像增强。
8. 证明中值滤波不是线性运算。
9. 将图像加入胡椒粉式的脉冲干扰，再以 3×3、5×5 及 7×7 的中值滤波器进行图像增强的处理。比较并讨论所得结果。
10. 将图 5.2-5 所示的梯度的模板运算作用在一图像上，显示物体边缘的效果并比较效果的好坏。
11. 对一带胡椒粉式的干扰的图像重复上题。比较这两题结果的差异。
12. 设计一实验，以验证 5.3-1 节所提到的振铃效应。


备忘录

第六章

图像恢复



图像恢复的主要目的是去改善一幅品质遭受恶化的图像。事实上，复原是一种过程，此种过程试图利用对恶化现象发生之前的了解，建立恶化的模型，再运用相反的过程来重建或恢复图像。

早期的数字图像恢复技术大多是以频率域的方法为基础。但是，本章则以代数方法为探讨重点。这种方法的优点是它能由相同的基本原理推导出许多恢复技术。虽然代数方法一般包括庞大的联立方程组的处理，但正如以下诸节所示，在某些条件下，可以把计算复杂度简化到与传统的频率域恢复技术相同的程度。

6.1 图像降质系统

6.1-1 基本定义

在了解图像恢复前，要了解图像降质系统。如图 6.1-1 所示，假设输入原始图像为 $f(m,n)$，经降质系统（或算子）H 作用后，再加入噪声干扰 $\eta(m,n)$，得到输出的降质图像为 $g(x,y)$，其中 $\eta(m,n)$ 为相加性的随机过程。如果不是相加性噪声，而是相乘性的噪声，可以用对数变换方式转化。图 6.1-1 的系统可表示成

$$g(m,n)=H[f(m,n)]+\eta(m,n) \tag{6.1-1}$$

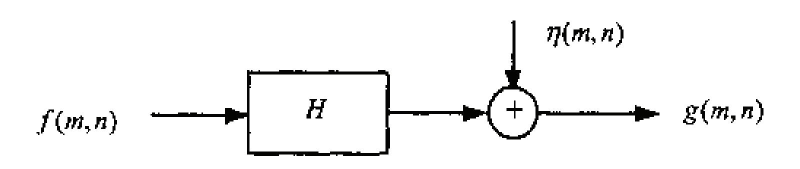


图 6.1-1 图像降质系统

在一般讨论中均假设降质系统 H 是线性时不变的 (参考 2.2 节有关 LSI 的定义)：

(1) H 是线性的，即在 $\eta(m,n)=0$ 时，满足下式

$$H[k_1 f_1(m,n)+k_2 f_2(m,n)]=k_1 H\left[f_1(m,n)\right]+k_2 H\left[f_2(m,n)\right] \tag{6.1-2}$$

式中 k_1 和 k_2 为常数。

(2) H 是空间 (或时) 不变的。意思是说如果系统的输入输出关系满足 $g(m,n)=H[f(m,n)]$，则对于任一个 $f(m,n)$ 和任一个常数 α 和 β 都有关系 $H[f(m-\alpha,\ n-\beta)]=g(m-\alpha,n-\beta)$。

在上述图像降质系统的模型下，图像恢复的问题变成是在已知 $g(m,n)$、H 及 $\eta(m,n)$ 的情形下，对原始图像 $f(m,n)$ 做估测。此外，一个 LSI 系统的特性可完全由其脉冲响应 $h(m,n)$ 来决定，亦即对任一输入，系统的输出等于此输入与脉冲响应的卷积。

6.1-2 降质系统的矩阵描述

为了以后方便描述，我们将降质系统以矩阵形式描述。首先考虑两个大小分别为 $M_1\times N_1$

以及 $M_2 \times N_2$ 的离散二维信号 $f(m,n)$ 与 $h(m,n)$，这两个信号的卷积所得的信号 $g'(m,n)$ 的大小为$(M = M_1+M_2-1)\times(N = N_1+N_2-1)$。为了方便矩阵的形成，我们将 $f(m,n)$ 与 $h(m,n)$ 以补零的方式分别扩增成大小均为 $M \times N$ 的信号 $f'(m,n)$ 与 $h'(m,n)$，亦即

$$f'(m,n)=\begin{cases} f(m,n), & 0 \leqslant m \leqslant M_1-1 \text{且} 0 \leqslant n \leqslant N_1-1 \\ 0, & M_1 \leqslant m \leqslant M-1 \text{或} N_1 \leqslant n \leqslant N-1 \end{cases}$$
$$h'(m,n)=\begin{cases} h(m,n), & 0 \leqslant m \leqslant M_2-1 \text{且} 0 \leqslant n \leqslant N_2-1 \\ 0, & M_2 \leqslant m \leqslant M-1 \text{或} N_2 \leqslant n \leqslant N-1 \end{cases} \tag{6.1-3}$$

因此，

$$g'(m,n)=\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} f'(i,j)h'(m-i,n-j) \tag{6.1-4}$$

其中 $m = 0, 1, 2,\cdots, M-1$；$n = 0,1,2,\cdots,N-1$。将 (6.1-4) 式以矩阵方式写成

$$\boldsymbol{g}=\boldsymbol{Hf} \tag{6.1-5}$$

其中

$$\boldsymbol{g}=\begin{bmatrix} g'(0,0) \\ g'(0,1) \\ \vdots \\ g'(M-1,N-1) \end{bmatrix}，\quad \boldsymbol{f}=\begin{bmatrix} f'(0,0) \\ f'(0,1) \\ \vdots \\ f'(M-1,N-1) \end{bmatrix} \tag{6.1-6}$$

$$\boldsymbol{H}=\begin{bmatrix} \boldsymbol{H}_0 & \boldsymbol{H}_{M-1} & \boldsymbol{H}_{M-2} & \cdots & \boldsymbol{H}_1 \\ \boldsymbol{H}_1 & \boldsymbol{H}_0 & \boldsymbol{H}_{M-1} & \cdots & \boldsymbol{H}_2 \\ \boldsymbol{H}_2 & \boldsymbol{H}_1 & \boldsymbol{H}_0 & \cdots & \boldsymbol{H}_3 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \boldsymbol{H}_{M-1} & \boldsymbol{H}_{M-2} & \boldsymbol{H}_{M-3} & \cdots & \boldsymbol{H}_0 \end{bmatrix}$$

其中

$$\boldsymbol{H}_i=\begin{bmatrix} h'(i,0) & h'(i,N-1) & h'(i,N-2) & \cdots & h'(i,1) \\ h'(i,1) & h'(i,0) & h'(i,N-1) & \cdots & h'(i,2) \\ h'(i,2) & h'(i,1) & h'(i,0) & \cdots & h'(i,3) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ h'(i,N-1) & h'(i,N-2) & h'(i,N-3) & \cdots & h'(i,0) \end{bmatrix} \tag{6.1-7}$$

依类似的扩展矩阵形成程序，我们将噪声 $\eta(m,n)$ 亦纳入系统中，而形成完整的降质系统的矩阵描述

$$\boldsymbol{g}=\boldsymbol{Hf}+\boldsymbol{n} \tag{6.1-8}$$

其中 $\boldsymbol{n}$ 代表由 $\eta(m,n)$ 的扩增信号 $\eta'(m,n)$ 所形成的噪声矩阵。

由 (6.1-7) 式可看出，$\boldsymbol{H}_i$ 本身为循环矩阵，又由 (6.1-6) 式可进一步看出 $\boldsymbol{H}$ 为方块循环矩阵。从实用的角度来看，矩阵 $\boldsymbol{H}$ 的小大为 $MN \times MN$，因此若 $M=N=512$，则 H 的大小为 $262\ 144 \times 262\ 144$，这对一般电脑而言要直接从 (6.1-8) 式计算出 $\boldsymbol{f}$ 并不实际。所幸，由于 $\boldsymbol{H}$ 是循环矩阵，我们可证明 (留做习题)

$$\boldsymbol{H}=\boldsymbol{W}\boldsymbol{D}\boldsymbol{W}^{-1} \text{ 或 } \boldsymbol{D}=\boldsymbol{W}^{-1}\boldsymbol{H}\boldsymbol{W} \tag{6.1-9}$$

其中 $\boldsymbol{W}$ 为 $\boldsymbol{H}$ 的特征向量所构成的矩阵，$\boldsymbol{D}$ 则为 $\boldsymbol{H}$ 的特征值所构成的对角矩阵。此外，$\boldsymbol{W}$ 的元素基本上是离散傅立叶变换的变换基底，而 $\boldsymbol{D}$ 的元素则为 $h'(m,n)$ 的离散傅立叶变换 $H(k,l)$。

设 $G(k,l)$ 为 $g'(k,l)$ 的傅立叶变换，即

$$G(k,l)=\frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} g'(m,n)\exp\left[-\mathrm{j}2\pi\left(\frac{mk}{M}+\frac{nl}{N}\right)\right] \tag{6.1-10}$$

则我们可证明 $\boldsymbol{W}^{-1}\boldsymbol{g}$ 相当于

$$\begin{bmatrix} G(0,0) \\ G(0,1) \\ \vdots \\ G(M-1,N-1) \end{bmatrix} \tag{6.1-11}$$

亦即 $\boldsymbol{W}^{-1}$ 相当于一算子，对后面运算所得的效果就是取其傅立叶变换，并将其系数堆叠成矩阵。结合 (6.1-8) 与 (6.1-9) 式我们可得

$$\boldsymbol{W}^{-1}\boldsymbol{g}=\boldsymbol{D}\boldsymbol{W}^{-1}\boldsymbol{f}+\boldsymbol{W}^{-1}\boldsymbol{n} \tag{6.1-12}$$

转成相对应的离散傅立叶变换，每个元素之间有如下关系

$$G(k,l)=H(k,l)F(k,l)+N(k,l) \tag{6.1-13}$$

其中 $k=0,1,2,\cdots,M-1$；$l=0,1,2,\cdots,N-1$。

(6.1-12) 式提供一个解 (6.1-8) 式的另一个好方法，即进行几个 $M \times N$ 的离散傅立叶变换就好，而不需要考虑 $MN \times MN$ 矩阵的计算问题。此外，若 M 与 N 为 2 的乘幂，则有很多现成的快速傅立叶计算法可进一步降低求离散傅立叶变换的计算量。

6.2 代数恢复方法

代数法的主要概念是在寻找一个 f 的估计 $\hat{f}$，使得事先定义的性能准则最小化。此处将采用简单的最小平方准则函数。做此选择的主要优点是对于推导出若干熟知的恢复法提供一个统一的结构。这些恢复法可分为有条件限制性及无条件限制性两类。

6.2-1　无条件限制性恢复

由 (6.1-8) 式可知，降质模型中的噪声项为

$$\boldsymbol{n} = \boldsymbol{g} - \boldsymbol{H}\boldsymbol{f} \tag{6.2-1}$$

此处 $\boldsymbol{n}$、$\boldsymbol{g}$、$\boldsymbol{f}$、$\boldsymbol{H}$ 均为矩阵向量形式，其维度分别为 $MN\times1$，$MN\times1$，$MN\times1$ 与 $MN\times MN$。

当缺乏对于 $\boldsymbol{n}$ 的任何了解时，一个有意义的准则函数是寻找这样一个 $\hat{\boldsymbol{f}}$，让所假定噪声项的范数尽可能地小，使得 $\boldsymbol{H}\hat{\boldsymbol{f}}$ 在最小平方意义下近似于 $\boldsymbol{g}$。换言之，我们想寻找一个 $\hat{\boldsymbol{f}}$，使得

$$\|\boldsymbol{n}\|^2 = \|\boldsymbol{g} - \boldsymbol{H}\hat{\boldsymbol{f}}\|^2 \tag{6.2-2}$$

为最小。上式中，按照定义

$$\|\boldsymbol{n}\|^2 = \boldsymbol{n}^{\mathrm{T}}\boldsymbol{n} \text{ 和 } \|\boldsymbol{g} - \boldsymbol{H}\hat{\boldsymbol{f}}\|^2 = (\boldsymbol{g} - \boldsymbol{H}\hat{\boldsymbol{f}})^{\mathrm{T}}(\boldsymbol{g} - \boldsymbol{H}\hat{\boldsymbol{f}}) \tag{6.2-3}$$

分别为 $\boldsymbol{n}$ 和 $(\boldsymbol{g} - \boldsymbol{H}\hat{\boldsymbol{f}})$ 的范数的平方。由 (6.2-2) 式可知此一问题相当于准则函数

$$J(\hat{\boldsymbol{f}}) = \|\boldsymbol{g} - \boldsymbol{H}\hat{\boldsymbol{f}}\|^2 \tag{6.2-4}$$

对于 $\hat{\boldsymbol{f}}$ 的最小化问题。除了要求它使 (6.2-4) 式为最小外，没有任何其他条件的限制。

使 (6.2-4) 式为最小很容易办到，亦即只要将 J 对 $\hat{\boldsymbol{f}}$ 微分，并令其结果等于零向量即得

$$\frac{\partial J(\hat{\boldsymbol{f}})}{\partial \hat{\boldsymbol{f}}} = \boldsymbol{0} = -2\boldsymbol{H}^{\mathrm{T}}(\boldsymbol{g} - \boldsymbol{H}\hat{\boldsymbol{f}}) \tag{6.2-5}$$

对 $\hat{\boldsymbol{f}}$ 求解 (6.2-5) 式，得出

$$\hat{\boldsymbol{f}} = (\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H})^{-1}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{g} \tag{6.2-6}$$

假设 $\boldsymbol{H}^{-1}$ 存在，则 (6.2-6) 式简化为

$$\hat{\boldsymbol{f}} = \boldsymbol{H}^{-1}(\boldsymbol{H}^{\mathrm{T}})^{-1}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{g} = \boldsymbol{H}^{-1}\boldsymbol{g} \tag{6.2-7}$$

6.2-2　有条件限制性恢复

在本节里，我们把最小平方恢复问题看做是在条件 $\|\boldsymbol{n}\|^2 = \|\boldsymbol{g} - \boldsymbol{H}\hat{\boldsymbol{f}}\|^2$ 的限制下，范数 $\|\boldsymbol{Q}\hat{\boldsymbol{f}}\|^2$ 所构成的函数的最小化问题，其中 $\boldsymbol{Q}$ 是作用于 $\hat{\boldsymbol{f}}$ 的一个线性算子。这种方法使恢复过程相当复杂，因为对于 $\boldsymbol{Q}$ 的不同选择会得到不同的解。

在最小化问题中，对于一个等式条件限制的加入，可以利用第二章中所提到的拉格朗日

乘数法加以处理。其步骤是把条件限制表示成$\alpha\left(\|\boldsymbol{g}-\boldsymbol{H}\hat{\boldsymbol{f}}\|^2-\|\boldsymbol{n}\|^2\right)$的形式，然后把它加到函数$\|\boldsymbol{Q}\hat{\boldsymbol{f}}\|^2$上。换言之，我们寻找一个$\hat{\boldsymbol{f}}$，使得准则函数

$$J(\hat{\boldsymbol{f}})=\|\boldsymbol{Q}\hat{\boldsymbol{f}}\|^2+\alpha\left(\|\boldsymbol{g}-\boldsymbol{H}\hat{\boldsymbol{f}}\|^2-\|\boldsymbol{n}\|^2\right) \tag{6.2-8}$$

为最小。式中α是一个常数，称为拉格朗日乘数。一旦加上条件限制之后，最小化问题就可以用通常的方法来求解。

将(6.2-8)式对$\hat{\boldsymbol{f}}$微分并令其结果等于零向量，得出

$$\frac{\partial J(\hat{\boldsymbol{f}})}{\partial \hat{\boldsymbol{f}}}=\boldsymbol{0}=2\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q}\hat{\boldsymbol{f}}-2\alpha\boldsymbol{H}^{\mathrm{T}}(\boldsymbol{g}-\boldsymbol{H}\hat{\boldsymbol{f}}) \tag{6.2-9}$$

对 (6.2-9) 式求解$\hat{\boldsymbol{f}}$，即得

$$\hat{\boldsymbol{f}}=(\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H}+\gamma\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q})^{-1}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{g} \tag{6.2-10}$$

其中$\gamma=1/\alpha$。这个等式必须调整到满足本问题的条件限制。

6.3　反滤波法

我们用 (6.2-7) 式的无条件限制结果开始推导图像恢复技术。利用 (6.1-9) 式将 (6.2-7) 式改写成

$$\hat{\boldsymbol{f}}=\boldsymbol{H}^{-1}\boldsymbol{g}=(\boldsymbol{W}\boldsymbol{D}\boldsymbol{W}^{-1})^{-1}\boldsymbol{g}=\boldsymbol{W}\boldsymbol{D}^{-1}\boldsymbol{W}^{-1}\boldsymbol{g} \tag{6.3-1}$$

式中$\boldsymbol{W}$为$\boldsymbol{H}$的特征向量所构成的矩阵，$\boldsymbol{D}$为$\boldsymbol{H}$的特征值所构成的对角矩阵。

将 (6.3-1) 式的两边乘以$\boldsymbol{W}^{-1}$，得到

$$\boldsymbol{W}^{-1}\hat{\boldsymbol{f}}=\boldsymbol{D}^{-1}\boldsymbol{W}^{-1}\boldsymbol{g} \tag{6.3-2}$$

利用 6.1-2 节的结果，我们将组成 (6.3-2) 式的元素写成下面形式

$$\hat{F}(k,l)=\frac{G(k,l)}{H(k,l)} \tag{6.3-3}$$

其中$k=0,1,2,\cdots,M-1$；$l=0,1,2,\cdots,N-1$。

(6.3-3) 式所表示的图像恢复法通常称为反滤波器 (inverse filter) 法。这个术语的由来是把$H(k,l)$看做一个“滤波器”函数，它乘上$\hat{F}(k,l)$便会产生退化图像$g(m,n)$的傅立叶变换。在 (6.3-3) 式中，$G(k,l)$除以$H(k,l)$形成了反滤波作用。当然，恢复图像是利用下面的关系式得到的

$$\hat{f}(m,n)=\mathcal{F}^{-1}\left[\hat{F}(k,l)\right]=\mathcal{F}^{-1}\left[G(k,l)/H(k,l)\right] \tag{6.3-4}$$

其中 $m=0,1,2,\cdots,M-1$；$n=0,1,2,\cdots,N-1$。(6.3-4) 式可用 FFT 算法来实现。

值得注意的是在 (6.3-4) 式中，如果 $H(k,l)$ 在 kl 平面上的任意区域内为零或变得非常小，那么在恢复过程中就会遇到计算上的困难。如果 $H(k,l)$ 的零点位于 kl 平面的少数几个点上，则在计算 $\hat{F}(k,l)$ 时一般可以忽略，而对恢复结果不会有明显的影响。

6.4　最小平方滤波器

在最小平方滤波器图像恢复法的推导过程中，我们首先定义 $\boldsymbol{R}_f$ 为 $\boldsymbol{f}$ 的相关矩阵，亦即

$$\boldsymbol{R}_f = E\{\boldsymbol{f}\boldsymbol{f}^{\mathrm{T}}\} \tag{6.4-1}$$

式中 E 表示代数期望值运算。$\boldsymbol{R}_f$ 的第 i 列与第 j 行的元素用 $E\{f_i f_j\}$ 表示，它是 $\boldsymbol{f}$ 的第 i 个和第 j 个元素之间的关系。因为 $\boldsymbol{f}$ 的元素为实数，所以

$$r_{ij}=E\{f_i f_j\}=E\{f_j f_i\}=r_{ji} \tag{6.4-2}$$

假设任意两像素的相关性只与像素间的距离有关，而与它们所在的位置无关，则 $\boldsymbol{R}_f$ 可用方块循环矩阵 $\boldsymbol{R}$ 来表示

$$\boldsymbol{R}=\begin{bmatrix} \boldsymbol{R}_0 & \boldsymbol{R}_{M-1} & \cdots & \boldsymbol{R}_1 \\ \boldsymbol{R}_1 & \boldsymbol{R}_0 & \cdots & \boldsymbol{R}_2 \\ \boldsymbol{R}_2 & \boldsymbol{R}_1 & \cdots & \boldsymbol{R}_3 \\ \vdots & \vdots & \vdots & \vdots \\ \boldsymbol{R}_{M-1} & \boldsymbol{R}_{M-2} & \cdots & \boldsymbol{R}_0 \end{bmatrix} \tag{6.4-3}$$

其中

$$\boldsymbol{R}_i=\begin{bmatrix} r_{i,0} & r_{i,N-1} & \cdots & r_{i,1} \\ r_{i,1} & r_{i,0} & \cdots & r_{i,2} \\ r_{i,2} & r_{i,1} & \cdots & r_{i,3} \\ \vdots & \vdots & \vdots & \vdots \\ r_{i,N-1} & r_{i,N-2} & \cdots & r_{i,0} \end{bmatrix} \tag{6.4-4}$$

因而由 6.1-2 节的讨论可知，我们可利用 $\boldsymbol{R}$ 的特征向量组成一个 $\boldsymbol{W}$ 矩阵对它们进行对角化

$$\begin{aligned} \boldsymbol{W} &= \boldsymbol{W}_M \otimes \boldsymbol{W}_N = \{w_M(i,m)\}\otimes\{w_N(n,k)\} \\ &= \left\{ \mathrm{e}^{-\mathrm{j}\frac{2\pi}{M}im} \right\} \otimes \left\{ \mathrm{e}^{-\mathrm{j}\frac{2\pi}{N}nk} \right\} \end{aligned} \tag{6.4-5}$$

式中 $i,m=0,1,2,\cdots,M-1$，且 $n,k=0,1,2,\cdots,N-1$，则 $\boldsymbol{R}_f=\boldsymbol{W}\boldsymbol{A}\boldsymbol{W}^{-1}$，其中对角矩阵 $\boldsymbol{A}$ 中的元素为相关矩阵 $\boldsymbol{R}_f$ 中诸元素的傅立叶变换。接着，我们定义 $\boldsymbol{R}_n$ 为 $\boldsymbol{n}$ 的相关矩阵，亦即

$$\boldsymbol{R}_n = E\{\boldsymbol{n}\boldsymbol{n}^{\mathrm{T}}\} \tag{6.4-6}$$

则同理可得 $\boldsymbol{R}_n = \boldsymbol{W}\boldsymbol{B}\boldsymbol{W}^{-1}$，其中对角矩阵 $\boldsymbol{B}$ 的元素为相关矩阵 $\boldsymbol{R}_n$ 中诸元素的傅立叶变换。

我们用 $S_f(k,l)$和$S_\eta(k,l)$ 表示 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 矩阵中各元素。因为 $\boldsymbol{R}_f$ 和 $\boldsymbol{R}_n$ 中的各元素是 $\boldsymbol{f}$ 和 $\boldsymbol{n}$ 各元素之间的相关函数，因此 $S_f(k,l)$和$S_\eta(k,l)$ 分别为 $f\,(m,n)$ 和 $\eta\,(m,n)$ 的功率谱密度。

我们选择线性算子 $\boldsymbol{Q}$ 满足以下关系

$$\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q} = \boldsymbol{R}_f^{-1}\boldsymbol{R}_n \tag{6.4-7}$$

代入 (6.2-10) 式得

$$\hat{\boldsymbol{f}} = (\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H} + \gamma\boldsymbol{R}_f^{-1}\boldsymbol{R}_n)^{-1}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{g} \tag{6.4-8}$$

利用 (6.1-9) 式及 $\boldsymbol{R}_f = \boldsymbol{W}\boldsymbol{A}\boldsymbol{W}^{-1}$ 及 $\boldsymbol{R}_n = \boldsymbol{W}\boldsymbol{B}\boldsymbol{W}^{-1}$ 的结果可得

$$\hat{\boldsymbol{f}} = (\boldsymbol{W}\boldsymbol{D}^*\boldsymbol{D}\boldsymbol{W}^{-1} + \gamma\boldsymbol{W}\boldsymbol{A}^{-1}\boldsymbol{B}\boldsymbol{W}^{-1})\boldsymbol{W}\boldsymbol{D}^*\boldsymbol{W}^{-1}\boldsymbol{g} \tag{6.4-9}$$

两边共乘 $\boldsymbol{W}^{-1}$，化简后可得

$$\boldsymbol{W}^{-1}\hat{\boldsymbol{f}} = (\boldsymbol{D}^*\boldsymbol{D} + \gamma\boldsymbol{A}^{-1}\boldsymbol{B})^{-1}\boldsymbol{D}^*\boldsymbol{W}^{-1}\boldsymbol{g} \tag{6.4-10}$$

可以看出括号内的矩阵是对角矩阵。利用 6.1-2 节所发展出的观念可将 (6.4-10) 式中的元素写成下列形式

$$\hat{F}(k,l) = \left[\frac{H^*(k,l)}{|H(k,l)|^2 + \gamma[S_\eta(k,l)/S_f(k,l)]}\right]G(k,l) \tag{6.4-11}$$

式中 $k = 0,1,2,\cdots,M-1$；$l = 0,1,2,\cdots,N-1$；$\gamma = 1/\alpha$。

由上式可推出几个特别情况：

(1) 当 $\gamma = 1$ 时、式中较大括号内的项即为 Wiener 滤波器。若 γ 为可变则称为参数 Wiener 滤波器。

(2) 在无噪声存在，即 $S_\eta(k,l) = 0$ 时、则变成

$$\hat{F}(k,l) = [1/H(k,l)]G(k,l) \tag{6.4-12}$$

这就是 6.3 节的反滤波器。

(3) 如果 $S_\eta(k,l)$ 和 $S_f(k,l)$ 为未知时 (这是实际上常碰到的状况)，则 (6.4-11) 式可表示为

$$\hat{F}(k,l) \approx \left[\frac{1}{H(k,l)} \cdot \frac{|H(k,l)|^2}{|H(k,l)|^2 + K}\right]G(k,l) \tag{6.4-13}$$

式中 K 是噪声对讯号的谱密度之比，它近似为一常数。此式可以使退化图像得到一定程度的恢复，但未必是最佳恢复。

6.5 限制性最小平方恢复

在上一节所列的最小平方法是一统计方法，因为其最佳化的准则是以图像和噪声函数的相关矩阵为基础。也就是说，用 Wiener 滤波器得到的结果在平均意义上是最佳的。但此处所研究的方法对每一幅所给的图像都是最佳的，并且只需要知道噪声的平均值和方差。

我们先考虑一维情况：对于一个离散函数 $f(m)$，m=0,1,2,…，在点 m 上的二阶导数可以近似地表示成如下形式

$$\frac{\partial^2 f(m)}{\partial m^2} \approx f(m+1) - 2f(m) + f(m-1) \tag{6.5-1}$$

故以该式为基础的准则是使 $(\partial^2 f / \partial m^2)^2$ 在 m 上最小，亦即

$$\text{minimize}\left\{\sum_m [f(m+1) - 2f(m) + f(m-1)]^2\right\} \tag{6.5-2}$$

或者用矩阵符号表示为

$$\text{minimize}\left\{\boldsymbol{f}^{\mathrm{T}}\boldsymbol{C}^{\mathrm{T}}\boldsymbol{C}\boldsymbol{f}\right\} \tag{6.5-3}$$

其中

$$\boldsymbol{C} = \begin{bmatrix} 1 & & & & & & \\ -2 & 1 & & & & & \\ 1 & -2 & 1 & & & & \\ & 1 & -2 & 1 & & & \\ & & & \ddots & & & \\ & & & & 1 & -2 & 1 \\ & & & & & 1 & -2 \\ & & & & & & 1 \end{bmatrix} \tag{6.5-4}$$

是一“平滑”矩阵，$\boldsymbol{f}$ 是 $f(m)$ 的取样值所形成的一个向量。

在二维状况下，我们考虑(6.5-1)式的直接推广。在此情况下，准则为

$$\text{minimize}\left[\frac{\partial^2 f(m,n)}{\partial m^2} + \frac{\partial^2 f(m,n)}{\partial n^2}\right]^2 \tag{6.5-5}$$

其中导函数是用下式来近似

$$\begin{aligned}\frac{\partial^2 f}{\partial m^2} + \frac{\partial^2 f}{\partial n^2} &\approx [2f(m,n) - f(m+1,n) - f(m-1,n)] \\ &\quad + [2f(m,n) - f(m,n+1) - f(m,n-1)] \\ &= 4f(m,n) - [f(m+1,n) + f(m-1,n) \\ &\quad + f(m,n+1) + f(m,n-1)]\end{aligned} \tag{6.5-6}$$

上式可以直接用计算机来实现。但是，利用 $f(m,n)$ 和下面的算子 $p(m,n)$ 的卷积亦可以完成同样的运算

$$p(m,n)=\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \tag{6.5-7}$$

此处使用扩增的 $f'(m,n)$ 和 $p'(m,n)$ 可以避免离散卷积过程中的重叠误差。我们以如下的方式构成 $p'(m,n)$

$$p'(m,n)=\begin{cases} p(m,n), & 0\leqslant m\leqslant 2 \text{ 且 } 0\leqslant n\leqslant 2 \\ 0, & 3\leqslant m\leqslant M-1 \text{或 } 3\leqslant n\leqslant N-1 \end{cases} \tag{6.5-8}$$

如果 $f(m,n)$ 的大小为 $A\times B$，则我们选择 $M\geqslant A+3-1$ 和 $N\geqslant B+3-1$，因为 $p'(m,n)$ 的大小为 3×3。

于是扩增函数的乘积为

$$g'(m,n)=\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f'(x,y)p'(m-x,n-y) \tag{6.5-9}$$

我们在此将平滑准则表示为矩阵形式。首先我们建立一个如下形式的方块循环矩阵

$$\boldsymbol{C}=\begin{bmatrix} \boldsymbol{C}_0 & \boldsymbol{C}_{M-1} & \boldsymbol{C}_{M-2} & \cdots & \boldsymbol{C}_1 \\ \boldsymbol{C}_1 & \boldsymbol{C}_0 & \boldsymbol{C}_{M-1} & \cdots & \boldsymbol{C}_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \boldsymbol{C}_{M-1} & \boldsymbol{C}_2 & \boldsymbol{C}_1 & \cdots & \boldsymbol{C}_0 \end{bmatrix} \tag{6.5-10}$$

式中每个子矩阵 $\boldsymbol{C}_j$ 是一个 $N\times N$ 循环矩阵，它是由 $p'(m,n)$ 的第 j 行所构成的。即

$$\boldsymbol{C}_j=\begin{bmatrix} p'(j,0) & p'(j,N-1) & \cdots & p'(j,1) \\ p'(j,1) & p'(j,0) & \cdots & p'(j,2) \\ \vdots & \vdots & \vdots & \vdots \\ p'(j,N-1) & p'(j,N-2) & \cdots & p'(j,0) \end{bmatrix} \tag{6.5-11}$$

因为 $\boldsymbol{C}$ 为方块循环矩阵，故可用 6.1 节中所定义的矩阵 $\boldsymbol{W}$ 进行对角化。亦即，

$$\boldsymbol{E}=\boldsymbol{W}^{-1}\boldsymbol{C}\boldsymbol{W} \tag{6.5-12}$$

式中 $\boldsymbol{E}$ 是一个对角矩阵，它的元素为

$$E(k,i)=\begin{cases} P\left(\left[\dfrac{k}{N}\right],\quad k \bmod N\right), & \text{如果 } i=k \\ 0, & \text{如果 } i\neq k \end{cases} \tag{6.5-13}$$

其中 $P(k,l)$ 是 $p'(m,n)$ 的二维傅立叶变换乘以因子 MN。

上述卷积运算与实现 (6.5-6) 式是等效的，所以 (6.5-6) 式的平滑准则取和 (6.5-3) 式相同的形式

$$\text{minimize}\left\{\boldsymbol{f}^{\mathrm{T}}\boldsymbol{C}^{\mathrm{T}}\boldsymbol{C}\boldsymbol{f}\right\} \tag{6.5-14}$$

式中 $\boldsymbol{f}$ 是一个 MN 维向量，$\boldsymbol{C}$ 的大小为 $MN\times MN$。若令 $\boldsymbol{Q}=\boldsymbol{C}$，并利用 $\|\boldsymbol{QF}\|^2=(\boldsymbol{Q}\boldsymbol{f})^{\mathrm{T}}(\boldsymbol{Q}\boldsymbol{f})=\boldsymbol{f}^{\mathrm{T}}\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q}\boldsymbol{f}$，则这个准则可以表示为

$$\text{minimize}\|\boldsymbol{Q}\boldsymbol{f}\|^2 \tag{6.5-15}$$

事实上，如果我们要求满足限制条件 $\|\boldsymbol{g}-\boldsymbol{H}\hat{\boldsymbol{f}}\|^2=\|\boldsymbol{n}\|^2$，则 (6.2-10) 式在 $\boldsymbol{Q}=\boldsymbol{C}$ 时得出最佳解

$$\hat{\boldsymbol{f}}=(\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H}+\gamma\boldsymbol{C}^{\mathrm{T}}\boldsymbol{C})^{-1}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{g} \tag{6.5-16}$$

利用与前一节所述的相同方法，上式成为

$$\hat{\boldsymbol{f}}=(\boldsymbol{W}\boldsymbol{D}^{*}\boldsymbol{D}\boldsymbol{W}^{-1}+\gamma\boldsymbol{W}\boldsymbol{E}^{*}\boldsymbol{E}\boldsymbol{W}^{-1})^{-1}\boldsymbol{W}\boldsymbol{D}^{*}\boldsymbol{W}^{-1}\boldsymbol{g} \tag{6.5-17}$$

两边都乘以 $\boldsymbol{W}^{-1}$ 并进行一些矩阵变换，上式简化为

$$\boldsymbol{W}^{-1}\hat{\boldsymbol{f}}=\left(\boldsymbol{D}^{*}\boldsymbol{D}+\gamma\boldsymbol{E}^{*}\boldsymbol{E}\right)^{-1}\boldsymbol{D}^{*}\boldsymbol{W}^{-1}\boldsymbol{g} \tag{6.5.18}$$

利用对角化对降质模型影响的概念，(6.5-18) 式可表示成

$$\hat{F}(k,l)=\left[\frac{H^{*}(k,l)}{|H(k,l)|^2+\gamma|P(k,l)|^2}\right]G(k,l) \tag{6.5-19}$$

其中 $k,l=0,1,2,\cdots,N-1$，$|H(k,l)|^2=H(k,l)^{*}H(k,l)$，并且已假设 $M=N$。注意到 (6.5-19)式与 6.4 节讨论的参数 Wiener 滤波器类似。(6.4-11) 式与 (6.5-19) 式之间最主要的区别是后者要估计噪声平均值和方差而不要求确切知道其他统计参数。(6.2-10) 式所列的一般公式要求调整 γ 使得限制条件 $\|\boldsymbol{g}-\boldsymbol{H}\boldsymbol{f}\|^2=\|\boldsymbol{n}\|^2$ 得以满足，同理 (6.5-19) 式所给的解也只有当 γ 满足这一条件时才是最佳的。要估算此一参数，有一个可用的迭代程序如下所述。

首先定义残余向量 $\boldsymbol{r}$ 为

$$\boldsymbol{r}=\boldsymbol{g}-\boldsymbol{H}\hat{\boldsymbol{f}} \tag{6.5-20}$$

用 (6.5-16) 式代替 $\hat{\boldsymbol{f}}$，得到

$$\boldsymbol{r}=\boldsymbol{g}-\boldsymbol{H}(\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H}+\gamma\boldsymbol{C}^{\mathrm{T}}\boldsymbol{C})^{-1}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{g} \tag{6.5-21}$$

(6.5-21)式指出，$\boldsymbol{r}$ 是 γ 的一个函数，可以证明

$$\phi(\gamma)=\boldsymbol{r}^{\mathrm{T}}\boldsymbol{r}=\|\boldsymbol{r}\|^2 \tag{6.5-22}$$

是γ的一个单调递增函数。我们调整γ使得

$$\|\boldsymbol{r}\| = \|\boldsymbol{n}\|^2 \pm \varepsilon \tag{6.5-23}$$

其中ε是一个准确度因子。可以很清楚地看出，如果$\|\boldsymbol{r}\|^2 = \|\boldsymbol{n}\|^2$，则从 (6.5-20) 式来看，将完全得到满足限制条件$\|\boldsymbol{g} - \boldsymbol{H}\hat{\boldsymbol{f}}\|^2 = \|\boldsymbol{n}\|^2$。

因为$\phi(\gamma)$ 是单调的，找到一个满足 (6.5-23) 式的γ并不困难。一个简单的方法是：

(1) 指定γ的一个起始值。

(2) 计算$\hat{\boldsymbol{f}}$和$\|\boldsymbol{r}\|^2$。

(3) 如果 (6.5-23) 式满足则停止；否则，如果$\|\boldsymbol{r}\|^2 < \|\boldsymbol{n}\|^2 - \varepsilon$，则在增加$\gamma$之后返回 (2)，如果$\|\boldsymbol{r}\|^2 > \|\boldsymbol{n}\|^2 + \varepsilon$，则在降低$\gamma$之后返回 (2)。

其他方法，如 Newton-Raphson 算法，可以用来改善收敛速度。

上述概念的实现必须对$\|\boldsymbol{n}\|^2$有些了解。$\eta'(m,n)$的方差为

$$\sigma_\eta^2 = E\left\{\left[\eta'(m,n) - \overline{\eta'}\right]^2\right\} = E\left\{\eta'^2(m,n)\right\} - \overline{\eta'}^2 \tag{6.5-24}$$

其中

$$\overline{\eta'} = \frac{1}{(M-1)(N-1)}\sum_m\sum_n \eta'(m,n) \tag{6.5-25}$$

是$\eta'(m,n)$的平均值。如果用取样平均来近似$\eta'^2(m,n)$的期望值，则 (6.5-24) 式变为

$$\sigma_\eta^2 = \frac{1}{(M-1)(N-1)}\sum_m\sum_n \eta'^2(m,n) - \overline{\eta'}^2 \tag{6.5.26}$$

总和项表示把阵列$\eta'(m,n)$中所有的$m = 0,1,2,\cdots, M-1$和$n = 0,1,2,\cdots, N-1$的项进行平方相加。这种运算就是简单的$\boldsymbol{n}^{\mathrm{T}}\boldsymbol{n}$积，根据定义它等于$\|\boldsymbol{n}\|^2$。所以 (6.5-26) 式化简为

$$\sigma_\eta^2 = \frac{\|\boldsymbol{n}\|^2}{(M-1)(N-1)} - \overline{\eta'}^2 \tag{6.5-27}$$

或

$$\|\boldsymbol{n}\|^2 = (M-1)(N-1)\left(\sigma_\eta^2 + \overline{\eta'}^2\right) \tag{6.5-28}$$

此式显示我们能利用噪声平均值和方差来确定限制值，而如果这些量是未知的，我们也可以实际估测出来。

限制最小平方恢复法可以总结如下：

(1) 选择γ的一个初始值，并利用 (6.5-28) 式得到$\|\boldsymbol{n}\|^2$的一个估计。

(2) 利用 (6.5-19) 式计算$\hat{F}(k,l)$。取$\hat{F}(k,l)$的傅立叶反变换得到$\hat{\boldsymbol{f}}$。

(3) 根据 (6.5-20) 式形成残余向量 $\boldsymbol{r}$，并计算 $\phi(\gamma)=\|\boldsymbol{r}\|^2$。

(4) 增加或减少 γ。

① $\phi(\gamma)<\|\boldsymbol{n}\|^2-\varepsilon$，根据上面给出的算法或其他近似方法 (Newton-Raphson 法) 增加 γ。

② $\phi(\gamma)>\|\boldsymbol{n}\|^2+\varepsilon$，根据一个适当算法减小 γ。

(5) 返回 (2) 并按顺序继续下去，一直到 (6) 为真。

(6) $\phi(\gamma)=\|\boldsymbol{n}\|^2\pm\varepsilon$，这里 ε 决定限制条件满足的精确度。停止估测过程。对这时的 γ 求得的 $\hat{\boldsymbol{f}}$ 便是恢复后的图像。

6.6　盲目图像恢复技术

6.6-1　简　介

假设一个受污染的图像 $g(m,n)$ 可用原始图像 $f(m,n)$ 与点扩散函数 (point-spread function, PSF) $h(m,n)$ 的卷积表示如下

$$g(m,n)=f(m,n)*h(m,n) \tag{6.6-1}$$

传统的线性图像恢复技术都是假设 PSF，即 $h(m,n)$ 是已知的，但在很多的实际情形中却不是如此。换言之，我们对 $h(m,n)$ 是 "盲目的(blind)"。这时，原始图像 $f(m,n)$ 必需直接借助 $g(m,n)$ 来估测。这种情形通常出现在事前图像信息取得有困难时，很可能是因为危险、花费庞大，或是根本无法取得。例如：首次拍摄的遥测图像或天文图像，其通道污染的过程难以即时得知；又如在诸多医学电视会议等应用的即时图像处理上，必须根据受污染的连续图像在线 (on-line) 估测其污染；还有在太空探险的任务上，可能因体积及重量等限制，被迫只能使用较低解析度的摄影机而获得有待图像恢复的较差图像；另外在 X 光取像上，增强 X 光的强度可提高图像品质，但可能对病人的健康有害，因而只能获得并不是太理想的图像。

典型的盲目图像恢复方块图如图 6.6-1 所示。原始图像 $f(m,n)$ 经过恶化模型 $h(m,n)$ 后，我们必须通过各种统计信息或物理特性来估测 PSF。最后，经过盲目恢复的算法估测出 $\hat{f}(m,n)$。我们的最终目的当然是要使估测的图像 $\hat{f}(m,n)$ 趋近于原始的图像 $f(m,n)$。

这一节主要是介绍五种盲目图像恢复技术：

① 零表面分离 (zero sheet separation)。

② 先验模糊辨认法 (a priori blur identification methods)。

③ ARMA 参数估测法。

④ 非参数式定型图像限制恢复技术。

⑤ 以高阶统计 (high order statistics) 为基础的非参数法。

并分析其适用范围、收敛性及复杂度等问题。

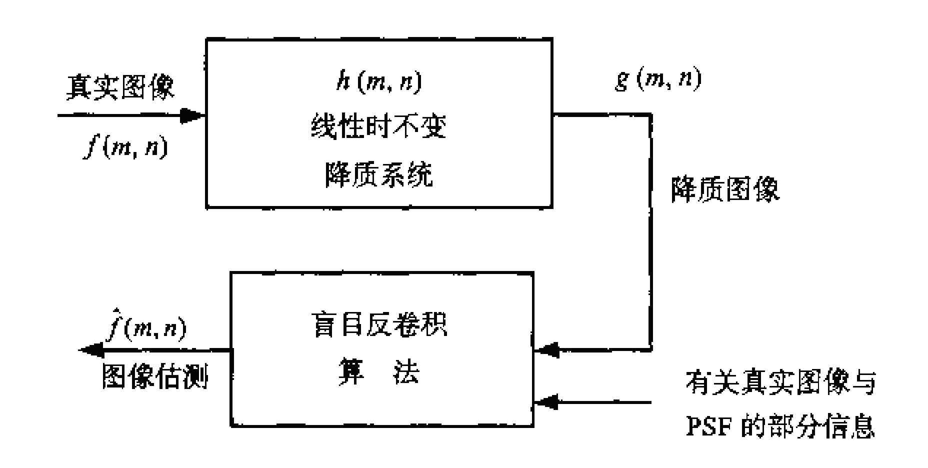


图 6.6-1　盲目图像恢复

6.6-2　方　法

● **零表面分离**

在多维 z 变换中有一个性质是说，一个 k 维序列的 z 变换的零点几乎都是连续，而且落在 $(2k-2)$ 维的一个*超曲面* (hypersurface) 上。假设有 r 个多维序列 $f_1, f_2, \cdots, f_r$ 的卷积、即

$$f_1 * f_2 * \cdots * f_r \tag{6.6-2}$$

则其 z 变换为

$$F_1 \times F_2 \times \cdots \times F_r \tag{6.6-3}$$

如果能将各 F_i 的零点所座落的超曲面分离，便可得各个 cf_i，其中 c 为一比例因子。这就是零表面分离的概念。

在不考虑相加性噪声的情况下，就二维图像恢复的问题而言，$k=r=2$，即

$$g(m,n) = f(m,n) * h(m,n) \tag{6.6-4}$$

其 z 变换为

$$G(z_1, z_2) = F(z_1, z_2)H(z_1, z_2) \tag{6.6-5}$$

由此可知多项式 $G(z_1, z_2)$是可分解的。因此，图像盲目恢复的问题相当于对二维多项式 $G(z_1, z_2)$ 的因式分解问题。而后者的问题有电脑可执行的算法来计算，不过相当繁琐。

零表面分离技术最有用之处是提供了洞察盲目恢复问题的能力、它可处理三维或更高维的问题。但其缺点是对噪声相当敏感，计算量也非常大。所以，这种方法在实现上有困难之处。

● **先验模糊辨识法**

这种方法最大的特色，也是其限制，是在做图像盲目恢复前、要先试图把 PSF 找出来。这种方法都对 PSF 有某些假设、或是已知其参数模型，例如：照相机水平晃动及焦距模糊所形成的 PSF。

有一种常用的先验模糊辨识法称频域零点法，其概念如下。对 (6.6-1) 式取傅立叶变换得

$$G(k,l)=F(k,l)\,H(k,l) \tag{6.6-6}$$

假设 PSF 有一个已知的参数形式，而且只要给出其频域零点 (即 $H(k,l)$) 的零点)，就可找出对应的唯一参数值。实际上常碰到的 PSF 可完全由其频域零点反应出来：

(1) 长度 $2d$ 的照相机水平移动模糊

$$h(m,n)=\begin{cases}0, & n\neq 0, \quad -\infty<m<\infty \\ \dfrac{1}{2d}, & n=0, \quad -d\leqslant m\leqslant d\end{cases} \tag{6.6-7}$$

此类 PSF 的频域零点落在与模糊方向直线上相距 $1/d$ 处。

(2) 有圆形孔隙的未对焦镜片系统

$$h(m,n)=\begin{cases}0, & \sqrt{m^2+n^2}>r \\ \dfrac{1}{\pi r^2}, & \sqrt{m^2+n^2}\leqslant r\end{cases} \tag{6.6-8}$$

所对应的频域零点落在以原点为中心且在 r 上近乎周期的同心圆上。

已知 $G(k,l)$ 的零点以及 PSF 的参数形式，盲目恢复的问题变成区别 $F(k,l)$ 的零点与 $H(k,l)$的零点的问题。有一个零点分离程序可用来解决此问题：一旦找到 $H(k,l)$ 的零点，就可估测 PSF 的参数值 (如果 PSF 是源自于照相机移动或不对焦的话，此即 (6.6-7) 式的 d 或 (6.6-8) 式的 r)。在获得 PSF 的估测后，就可使用传统的图像恢复法来估测出原始图像。

上述方法并未考虑相加性噪声存在时的情况，此噪声通常会使 $G(k,l)$ 的零点变得不明显，因此本方法比较适合于高信噪比的图像。本方法是普遍来用成功的一种方法，主要原因是其计算简单而且很可靠 (不会有其他方法因迭代无法收敛的状况发生)。其主要限制是必须知道 PSF 的参数形式。此外，在 X 光或天文图像的恢复问题上，因为其 PSF 通常是高斯的形式，使得 $H(k,l)$的频域零点不存在，此时就得用其他方式了。

● **ARMA 参数估测法**

ARMA 参数估测方法是把原始图像模式化成自回归 (autoregressive, AR) 过程，把 PSF 模式化成移动平均 (moving average, MA) 过程，模糊图像就表示成自回归移动平均 (autoregressive moving average, ARMA) 过程。原始图像 $f(m,n)$在 AR 模式里表示成

$$f(m,n)=\sum_{\substack{(k,l)\in R_a \\ (k,l)\neq(0,0)}} a(k,l)f(m-k,n-l)+v(m,n) \tag{6.6-9}$$

其中 $v(m,n)$是模式化过程的错误，$a(k,l)$是 AR 模式的系数，R_a 是 (k,l) 的范围。

如以矩阵—向量表示可写成

$$\boldsymbol{f}=\boldsymbol{A}\boldsymbol{f}+\boldsymbol{v} \tag{6.6-10}$$

降质图像 $g(m,n)$在 ARMA 模式里表示成

$$g(m,n)=\sum_{(k,l)\in R_h}h(k,l)f(m-k,n-l)+\eta(m,n) \tag{6.6-11}$$

其中 $\eta(m,n)$ 是可加性噪声，R_h 是 (k,l) 的范围。如以矩阵—向量表示可写成

$$\boldsymbol{g}=\boldsymbol{Hf}+\boldsymbol{n} \tag{6.6-12}$$

将 (6.6-10) 式代入 (6.6-12) 式得

$$\boldsymbol{g}=\boldsymbol{H}(\boldsymbol{I}-\boldsymbol{A})^{-1}\boldsymbol{v}+\boldsymbol{n} \tag{6.6-13}$$

其中 $\boldsymbol{I}$ 为单位矩阵。整个 ARMA 模式化过程的方块图如图 6.6-2 所示，其中大写字母代表其小写字母的 z 变换。

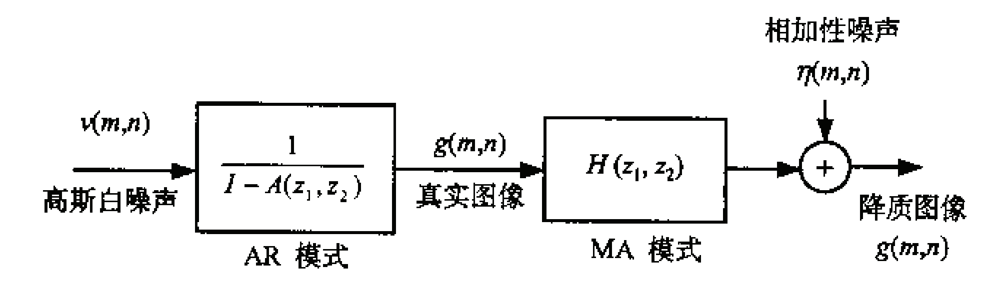


图 6.6-2　ARMA 模式化方块图

整个盲目恢复的问题现在变成从 $g(m,n)$ 中要估测出 $a(k,l)\in R_a$ 以及 $h(k,l)$, $(k,l)\in R_h$。一旦 $h(k,l)$ 决定后，传统的线性图像恢复法就可用来估测原图像。

实际上用 (6.6-13) 式估测 $a(k,l)$ 与 $h(k,l)$ 的困难包括高计算复杂度 (PSF 占较大范围时)，估测算法的不稳定性以及非惟一解等问题。因此也不是所有形式的 PSF 都适合用本方法，但像 (6.6-7) 及 (6.6-8) 式的 PSF 则没有问题。

ARMA 模式在二阶统计特性下有两种较常用的方法：

① 最大可能 (maximum-likelihood, ML) 法

② 广义交叉确认 (generalized cross-validation, GCV) 法

ML 法试图从 $g(m,n)$ 中估测出下列参数

$$\theta=\{\{h(k,l)\},\{a(k,l)\},\sigma_n^2,\sigma_v^2\} \tag{6.6-14}$$

其中 σ_n^2 与 σ_v^2 分别为 $\eta(m,n)$ 与 $v(m,n)$ 的方差。我们要的一个参数估测 $\hat{\theta}$ 是使接收到所观察而得图像的机会或可能性 (likelihood) 最大者，可写成

$$\hat{\theta}_{k,l}=\arg\{\max_{\theta\in\Theta}L(\theta)\}=\arg\{\max_{\theta\in\Theta}\log p(g;\theta)\} \tag{6.6-15}$$

其中 $L(\theta)$ 代表 θ 的 log-likelibood 函数，Θ 代表 θ 内各元素的范围，而 $p(g;\theta)$ 则是已知 θ 时的概率密度函数。

在 η 与 v 均假设为零平均高斯过程的情况下，可得到一个恰当的 $p(g;\theta)$。(6.6-15) 式的问题形成一种非线性最优化的问题，目前存在有许多方法，例如梯度为基础的方法、期望值为最大值的技术、估测误差为基础的方法，以及最小平方法等。

GCV 法是在数据分析上常采用的方法，其原理简述如下。首先数据分成二组。一组为估

测集，另一组为验证 (validation) 集。估测集是用来估测模型的参数值或某种假设，验证集则是验证这些估测或假设的表现情形。为了要更充分利用现有的数据、GCV 使所有数据都可扮演估测与验证的角色。

其做法是将数据分成 M 组。除了其中一组外，其他数据都用来验证某个参数或假设，记录对此组的验证误差。接着挑另一组，重复上述操作直到每一组都轮完为止，将各组的验证误差取平均就是对某个参数或假设的整体验证误差。

现在将 GCV 法用在图像恢复上。首先将降质图像 g 的数据分成 Mg 组，其中 Mg 代表图像 g 的像素个数。除一个像素外，其他像素都用来做图像恢复 (例如以 ML 法估测的 $h(k,l)$ 及 $a(k,l)$ 等)，接着将恢复后图像再以 $h(m,n)$ 做降质以预测该图像，记录估测误差。对其他像素重复上述动作以得到平均的验证误差。

上述 ML 和 GCV 的方法对相加性噪声都有良好的抗拒力，因其推导过程已把噪声考虑在内。这二种方法的主要限制为参数量变大时计算会有问题，且在做最优化过程时也有收敛到*局部极小点* (local minima) 的问题。

● **非参数式定型图像限制恢复技术**

这种方法不需要建立原始或模糊图像的参数模型，故其适用范围较广。下面针对常被使用的三种算法做讨论。

(1) *迭代盲目去卷积法*(Iterative Blind Deconvolution, IBD)

此方法最普遍采用的是以快速傅立叶变换为基础的算法，其方块图如图 6.6-3 所示。

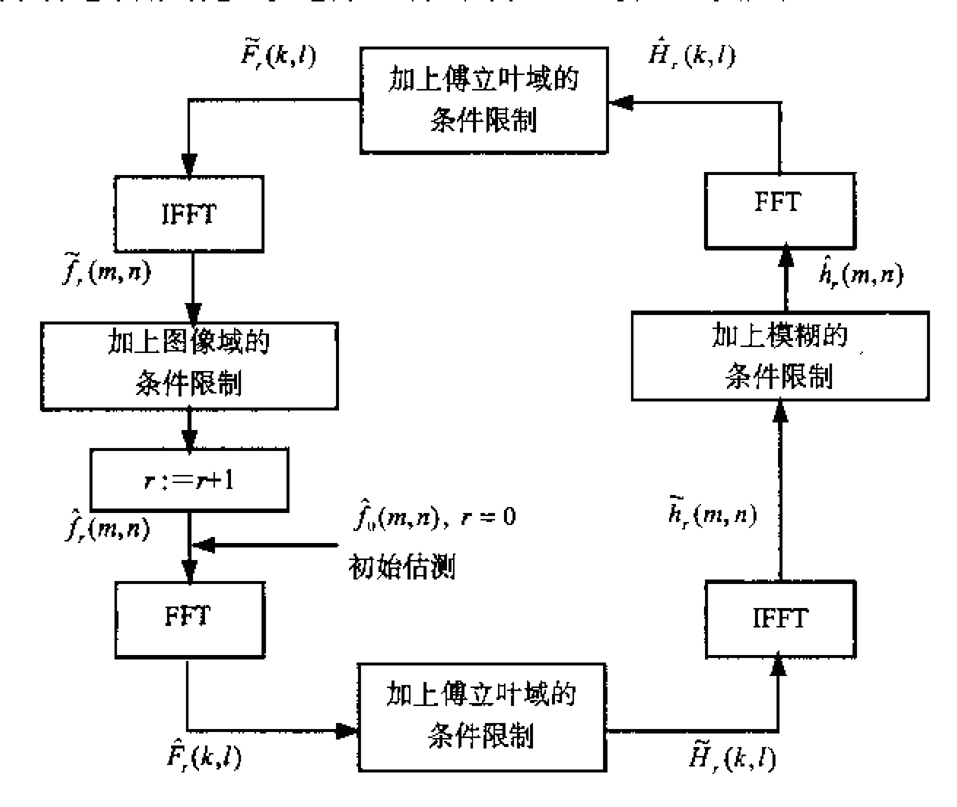


图 6.6-3　迭代盲目去卷积法

一开始我们假设一初始图像，接着就在图像与频率域间交替变换运算，并在每个过程中将

限制条件考虑进来。所加入的条件限制是根据对手边有关图像及 PSF 的信息，例如图像限制的条件包括将在所考虑的某物体图像的区域范围内 (region of support) 负值的像素值以零取代，以及范围外的非零像素以背景像素值取代等。在第 r 次迭代时，傅立叶域的条件限制可写成

$$\widetilde{H}_r(k,l)=\frac{G(k,l)\hat{F}^*_{r-1}(k,l)}{\left|\hat{F}_{r-1}(k,l)\right|^2+\dfrac{\alpha}{\left|\widetilde{H}_{r-1}(k,l)\right|^2}} \tag{6.6-16}$$

$$\widetilde{F}_r(k,l)=\frac{G(k,l)\hat{H}^*_{r-1}(k,l)}{\left|\hat{H}_{r-1}(k,l)\right|^2+\dfrac{\alpha}{\left|\widetilde{H}_{r-1}(k,l)\right|^2}} \tag{6.6-17}$$

其中 α 是相加性噪声的能量，必须慎选之才会有较佳的结果。

这种 IBD 的方法有较小的计算量，并且对噪声有好的抗拒力。主要的缺点是不确保迭代会收敛。另外，初始图像对结果会有相当大的影响。

(2) 模拟退火算法

MaCallum 的模拟退火 (simulated annealing, SA) 算法的主要步骤是将以下的成本函数最小化

$$J\left(\hat{f}(m,n),\hat{h}(m,n)\right)=\sum_{\forall(m,n)}\left[\hat{f}(m,n)*\hat{h}(m,n)-g(m,n)\right]^2 \tag{6.6-18}$$

此方法被称为模拟退火的原因是温度参数 T_k 必需缓慢减少才会使成本函数至整体最小值 (global minimum)，否则可能收敛至局部极小值 (local minimum)。这种情况就类似液态金属冷却时，温度必需缓慢下降才会使原子停留在稳定能阶。如果急速冷却，可能使原子停留在次佳 (suboptimal) 的能量状态。

SA 算法是很可靠的方法，它对抵抗噪声方面也有不错的表现。主要的缺点是对于一个一般大小的图像而言要获得好结果所需的计算量非常庞大。

(3) NAS-RIF 算法

NAS-RIF (nonnegativity and support constraints recursive inverse filtering) 算法的一个主要优点是 PSF 不必局限于有限范围。其方块图如图 6.6-4 所示，其中 $u(m,n)$ 为一自适性有限脉冲响应 (FIR) 滤波器，NL 代表一非线性滤波器，而误差信号 $e(m,n)$则通过像共轭梯度最小化法或最大陡降法等最优化方法来调整 $u(m,n)$ 的滤波器系数。有许多非线性滤波器可用，例如

$$\hat{f}_{NL}(m,n)=\begin{cases}\hat{f}(m,n), & 若(m,n)\in D_{sup}\\ L_B, & 其他情况\end{cases} \tag{6.6-19}$$

其中 D_{sup} 代表所考虑的物体图像范围内的区域，L_B 则是背景像素的色彩或灰度值。

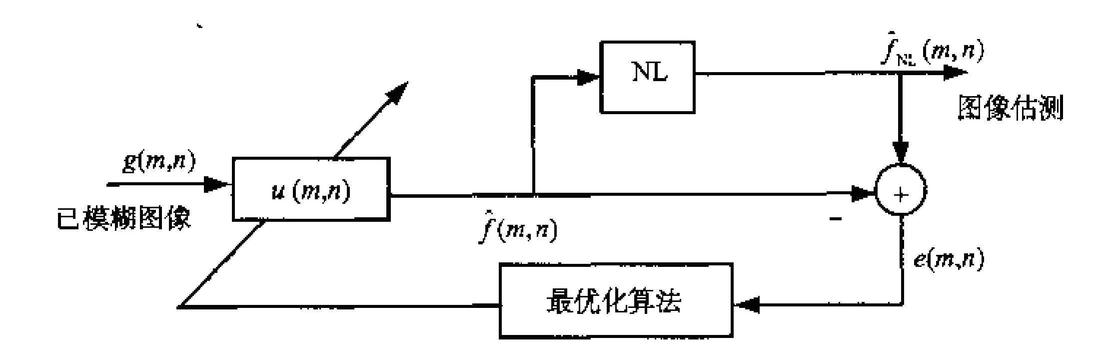


图 6.6-4　NAS-RIF 算法方块图

在最优化算法上可考虑成本函数如下

$$
\begin{aligned}
J(u) &= \sum_{\forall(m,n)} e^2(m,n) + \gamma\left[\sum_{\forall(m,n)} u(m,n) - 1\right]^2 \\
&= \sum_{(m,n)\in D_{sup}} \hat{f}^2(m,n)\left[\frac{1-\operatorname{sgn}(\hat{f}(m,n))}{2}\right] \\
&\quad + \sum_{(m,n)\in \bar{D}_{sup}}\left[\hat{f}(m,n) - L_B\right]^2 + \gamma\left[\sum_{\forall(m,n)} u(m,n) - 1\right]^2
\end{aligned}
\tag{6.6-20}
$$

其中 $\bar{D}_{sup}$ 是 D_{sup} 之外的区域，变量 γ 则是只有在 L_B 为零（全黑背景）时才有非零的值。由于上式的成本函数是凸状 (convex)，所以这个算法保证收敛到整体最小值，而没有掉入局部极小值的困境。

● **以高阶统计为基础的非参数法**

这个方法类似前述 NAS-RIF 算法，它适合处理纹理 (texture) 图像。其方块图如图 6.6-5 所示。其必须最小化的成本函数反应出图像的非高斯特性。

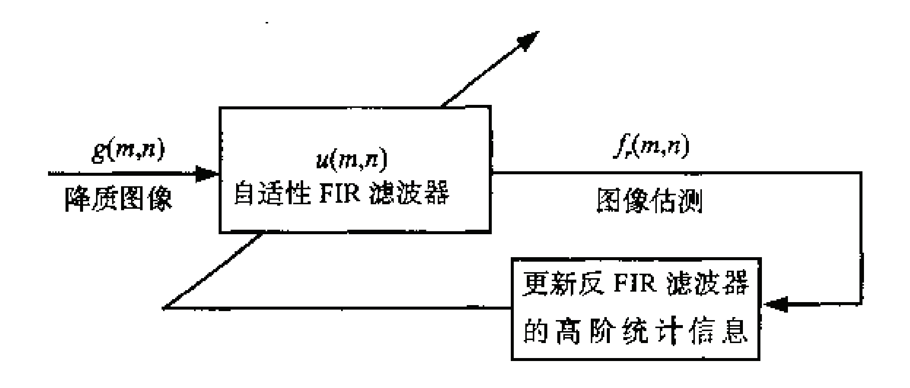


图 6.6-5　以高阶统计为基础的非参数法方块图

这种方法的主要优点是能辨别非最小相位 (non-minimum phase) 的 PSF，并且也有一定的抗干扰能力。其主要的限制是原始图像必需要模式化成非高斯概率分布，另一缺点是在最小化过程中可能掉入局部极小点。

习　　题

1. 考虑一线性时不变的图像降质系统，其脉冲响应为

$$h(x-\alpha, y-\beta)=\exp\left\{-(x-\alpha)^2(y-\beta)^2\right\}$$

假设系统的输入图像是位于 $x=\alpha$ 处，宽度无限小的一条线，其模型为

$$f(x,y)=\delta(x-a)$$

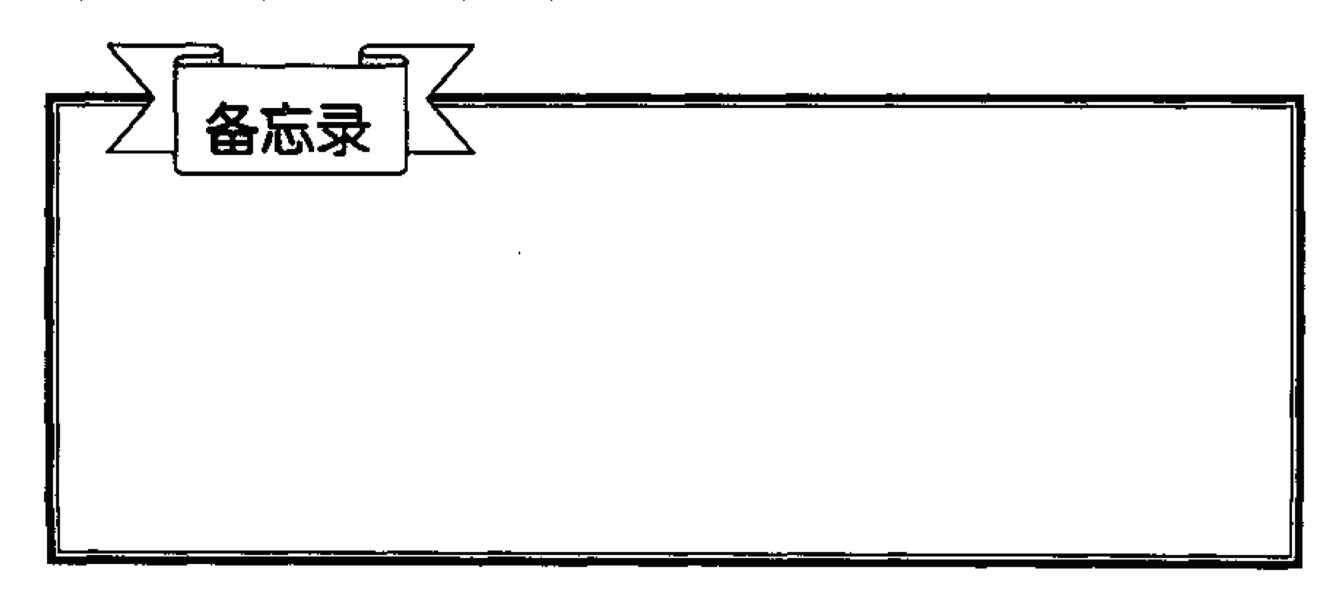
假设没有噪声，试求输出图像 $g(x,y)$。

2. 本题是有关*交互式* (interactive) 图像恢复的问题。首先将一幅图像加入一正弦 (或余弦)干扰，显示此幅干扰后的图像。看出正弦 (或余弦)干扰的特征了吗？假设你不知道干扰的振幅及频率，试以交互式的方式设计一带阻滤波器将该干扰滤掉。显示两个不当的设计以及一正确设计的结果，以便加以比较。
3. 设一点扩散函数 PSF 所形成的矩阵为 $\{h(m,n)\}$，其中

$$h(m,n)=\begin{cases}1/9, & m,n=0,1,2\\ 0, & \text{其他情况}\end{cases}$$

f 则在 $m,n=0,1,2,3$ 下有定义，试写出如 (6.1-3) 至 (6.1-7) 式的形式。

4. 证明 (6.1-9) 式成立。
5. 根据 (6.1-9) 式找出伴随第 3 题的扩散矩阵 $\{h(m,n)\}$ 的矩阵 $\boldsymbol{W}$ 与 $\boldsymbol{D}$。
6. 以第 3 题为例，验证 $\boldsymbol{W}^{-1}$ 相当于一取傅立叶变换的算子。
7. 设计一简单实验，可看出对降质图像实现最小平方恢复法的效果并实际实现之。
8. 假设一图像中有两个独立物体，其各别的范围设为已知。经某个图像模糊系统 (例如 (6.6-7) 及 (6.6-8)式) 及加入已知的噪声后得到某个降质图像，此时各物体的范围由于模糊的关系会扩大而超出原物体的范围。试设计一简单实验，以迭代盲目去卷积法实现图像恢复，并讨论实际上碰到的困难。



第七章

图像压缩



许多问题的产生是由于资源有限，而为了突破各种的限制，自然就有各种解决的方案。数据压缩可以说是在存储及传输方面的限制之下所发展出来的。

对图像压缩的研究可追溯至数十年前。最初研究的重点是放在减少图像传输带宽的模拟方法上，称为*带宽压缩* (bandwidth compression) 处理。由于数字计算机的出现及其后先进的集成电路的发展，促使研究注意力从模拟转向数字的压缩方法。20世纪40年代，C. E. Shannon等人首先从概率观点来看传输与压缩，奠定了图像压缩的理论基础。近年来，随着若干个重要的图像压缩国际标准的通过，促进了理论的实际应用，使图像压缩领域出现重大进展。

图像压缩的目的是减少表示数字图像所需的数据量。减少数据量的基本概念是去除重复多余的数据，而此冗余性又与数据间的相关性成正比。从数学角度来看，压缩是将相关性高的像素阵列变成统计上较无关的数据集的一种变换。此种变换使用在图像储存和传输之前。之后，被压缩的图像再被解压缩以重建原来的图像或它的近似值。本章将介绍图像数据的冗余性、图像品质的评估及各种常用的图像压缩法。

7.1 数据编码与数据压缩

在数据大量流通的今天，压缩所扮演的角色也日益重要。由于压缩使数据量减少，进而使数据传输时间能降低，让传输的效率更为提高。在传输带宽有限，扩充不易以及增加传输设备将提高成本的考虑下，压缩提供了另外一种解决方案。

压缩也易于产生保密的效果。由于压缩过程对于数据重新编码，改变原有数据的外貌，接收的对方要是没有相对应的解压缩程序就难以恢复。即使有意要破解，因为原有数据的许多特性已经改变，故较不容易被揣测出来。

一个数据压缩系统会包含两个搭配的子系统，一为*压缩器* (compressor) 或是*编码器* (encoder)，另一为*解压器* (decompressor) 或是*解码器* (decoder)。其关系如图 7.1-1 所示，其中 A 代表压缩前的数据，B 代表压缩后的结果，A′则代表 B 经过解压缩过程得到的结果。一个理想的数据压缩系统将会满足下列两个特性：

(1) $A \fallingdotseq A'$

(2) size (B) < size (A)

其中 $A \fallingdotseq A'$代表解压缩后的数据和原始的数据间不能有太大的差距，size(B) < size(A) 表示压缩后的数据所需的储存空间必须小于原始数据所需的储存空间，以节省储存空间或减少传输时间。size (A)/size(B)可定义成*压缩率* (compression ratio)，表示压缩的程度。

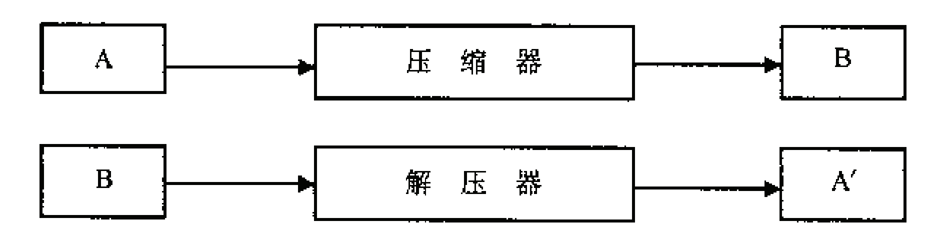


图 7.1-1 数据压缩系统方块图

7.1-1　数据相关性

数据压缩一词是指按所要求的信息品质去缩减数据量的一种处理。我们必须分清楚数据与信息之间的区别。数据是指传递信息的载体，同样的信息可以用不同量的数据来代表。假如我们有一个喜欢喋喋不休的人与另一个总是讲话简短扼要的人来讲同一个故事，这里我们感兴趣的信息是故事，而言语便是用来表达信息的数据。如果这两个人分别用他们不同数量的语言来叙述同一个故事，这就产生了同一故事的不同版本，其中至少有一个版本含有非必要的数据。这些数据 (或叫言语) 不是提供一些无关紧要的信息就是简单重复一些早已知道的东西。这就是所谓数据的*冗余性* (redundancy)。

对于数字图像压缩，有三种基本的数据冗余性可以运用，即：*编码* (coding) 冗余性、*像素间* (interpixel) 冗余性，及视觉 (psychovisual) 冗余性。消除或缩减上述三种冗余性的一种或多种，就可以得到数据压缩的效果。以下分别叙述这三种冗余性。

- **编码冗余性**

在一图像中，各个灰度出现的机率往往不完全相同，若不考虑此特性，一律采用自然二进制码来表示，则往往会有编码冗余性的产生。为了使编码更有效率，因此对出现概率较大的灰度分配以较少的位长度，以达到数据压缩的目的。此过程称为*可变字长编码* (variable-length coding)。

- **像素间冗余性**

通常将原始图像信号通过某种变换来解除图像信号间的相关性，亦即除去与图像中像素间相关性直接有关的冗余性。因为任一指定的像素灰度值皆可由邻近的像素灰度值作一适当的预测，因此各像素所携带的信息量相对减少。为了减少像素间的相关性，通常通过变换或称为*映射* (mapping) 来完成，例如第四章中所讨论过的 DCT、FFT 及 DWT 等，而原始的图像可以由这些变换的数据通过其反变换再完全重建，因此这种映射通常为可逆的。

- **视觉冗余性**

所谓视觉冗余性是指可以被删去而对图像的主观品质不会造成太大影响的冗余性。在一般情况下我们不会对个别像素作定量分析以取得图像信息，而是寻找一些诸如边界或纹理的特征，并将它们组成可辨识的群体，以便大脑完成对图像的理解。由于某些数据对正常视觉观察不是那么重要，因此得以被删除，这个过程称为量化处理。而视觉冗余性被删除，造成某些信息的损失，因此量化过程并非一可逆的过程，因而导致所谓有损耗的数据压缩，详见 7.5 节。

7.1-2　保真度准则

如前所述，由于视觉冗余性的删除会导致视觉信息的损失，因而重要的信息也可能被丢失，因此需要一个可衡量信息丢失的准则。此准则通常可分两大类：*客观保真度准则*

(objective fidelity criterion) 以及主观保真度准则 (subjective fidelity criterion)。

当信息损失的程度可以表示成原始输入图像与经过压缩及解压缩的输出图像的函数时，这种量测就是客观的保真度准则。输入与输出图像间的均方根误差即为一例。假设 $f(m,n)$ 表示输入图像，则将此图像经过压缩与解压缩所得的图像 $\hat{f}(m,n)$ 代表 $f(m,n)$ 的估测值或近似值。对任意的 m 与 n 值，$f(m,n)$ 与 $\hat{f}(m,n)$ 之间的误差 $e(m,n)$ 可以定义为

$$e(m,n)=\hat{f}(m,n)-f(m,n) \tag{7.1-1}$$

故整幅图像误差平方的总和为

$$\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}[\hat{f}(m,n)-f(m,n)]^2 \tag{7.1-2}$$

此处图像的大小为 $M\times N$。因此，$f(m,n)$ 与 $\hat{f}(m,n)$ 之间的均方根误差 e_{rms} 便是误差的平方在 $M\times N$ 阵列内取平均值的平方根，即

$$e_{rms}=\left[\frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}[\hat{f}(m,n)-f(m,n)]^2\right]^{\frac{1}{2}} \tag{7.1-3}$$

与客观保真度准则息息相关的是经过压缩与解压缩图像的均方信噪比 (mean-square signal-to-noise ratio)。如果把 $\hat{f}(m,n)$ 看成为原始图像 $f(m,n)$ 与噪声信号 $e(m,n)$ 之和，则输出图像的均方信噪比用 SNR_{ms} 表示为

$$\mathrm{SNR}_{ms}=\frac{\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\hat{f}(m,n)^2}{\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\left[\hat{f}(m,n)-f(m,n)\right]^2} \tag{7.1-4}$$

信噪比的均方根值，用 SNR_{rms} 表示，则可由上式取平方根而得。

另一种文献上常采用的客观保真度称为峰值信噪比 (peak SNR, PSNR)，其定义是将 (7.1-4)式的分子整个以 I_{max}^2 代换之，其中 I_{max} 是图像所容许的最大值，例如对一个 8 位的图像 $I_{max}=255$。此外，一般以分贝 (dB) 计算，因此必须再取 10 为底的对数，再乘以 10 才是最后所求。

虽然客观保真度准则提供了一种简单且方便的估算信息损失的方法，但大多数的解压缩图像的最终观察者还是人。因此，用观察者主观评估来衡量图像品质往往更为合适。其作法为：选一部分适当的观察者代表，让他们对典型的解压缩图像评分，然后取平均分数。评分可以用一种绝对等级尺度或对 $f(m,n)$ 与 $\hat{f}(m,n)$ 做出对照比较。表 7.1-1 表示一种可用的绝对等级尺度。对照比较则可应用尺度如 {-3, -2, -1, 0, 1, 2, 3} 分别代表主观评价{很差，较差，稍差，相同，稍好，较好，很好}。不论哪种情况的评分，都可说是以主观保真度为准则的。

表 7.1-1　电视画质的等级尺度

得　分	等　级	说　　明
1	优　秀	要求图像有极高品质
2	好　的	提供可欣赏的高品质图像、干扰不讨厌
3	及　格	图像品质可接受、干扰不讨厌
4	勉强及格	图像品质低，干扰有些讨厌
5	劣 等 的	图像很差，但尚能看，干扰明显
6	不能用的	图像差到不能看

7.2　图像压缩模型

在上一节中我们讨论了压缩一幅图像表示所需的三种数据冗余性。运用这些冗余性可建立一个实用的图像压缩系统。本节将考虑此系统的所有特性并以一个通用的模型来表示。

如图 7.2-1 所示，一个压缩系统有两个主要部分，即编码器与解码器。将输入图像 $f(m,n)$ 送进编码器后，编码器便根据输入数据产生一组符号。经过信道 (channel) 传输后送至解码器，由解码器输出一重建图像 $\hat{f}(m,n)$ 。一般说来，$\hat{f}(m,n)$ 可以和 $f(m,n)$ 完全相同或有所不同。若完全相同，则该系统是无失真的或信息保持的；反之，则重建图像会有一定程度的失真。

编码器由信源编码器与信道编码器所组成。前者删去输入冗余性，后者则增强信源编码器输出的抗干扰能力。解码器则包括信道解码器及与之相连结的信源解码器。如果在编码器与解码器之间的信道是无噪声 (不易发生错误) 的，则信道编码器与解码器可省去，于是编码器与解码器就简化成为信源编码器与解码器了。

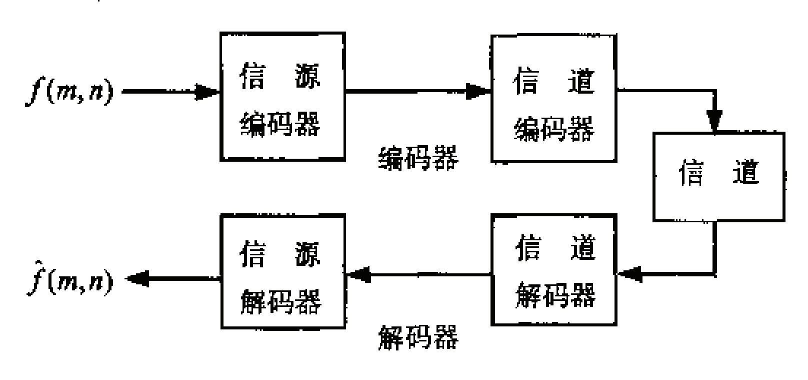


图 7.2-1　通用的压缩系统模型

7.3　信息论基础

7.3-1　信息度量

信息论的基本前提是信息的产生可用一个符合直觉的方法来度量，其中一个成功的度量

方式是以概率的模型来表示。设有一个随机事件 E，它的发生概率为 $P(E)$，则称它包含有

$$I(E)=\log\frac{1}{P(E)}=-\log P(E) \tag{7.3-1}$$

单位的信息。量 $I(E)$常称为 E 的自信息量 (self-information)。一般而言，由事件 E 所引起的自信息量与 E 的概率有倒数的关系。如果 $P(E)=1$(即事件总是发生)，$I(E)=0$，即它不提供信息。这是因为该事件不带有任何不确定性，当告知该事件已经发生时，并不提供什么信息。但是，如果 $P(E)=0.99$，当告知该事件已经发生时，还是传递了一点信息。若告知 E 并没有发生，则传递了更多信息，因为这个结果是不大可能的。

上式中对数的基底决定度量信息的单位。如果对数的基底为 r，则信息的度量为 r 进制单位。如果选择 2 为基底，则信息量的单位为位 (bit)。

7.3-2 信 道

考虑图 7.3-1 所示的简单信息系统。设信息源的可能符号为 $A=\{a_1,a_2,\cdots,a_J\}$，其中对应产生的概率集为向量 $\mathbf{z}=[P(a_1), P(a_2),\cdots, P(a_J)]^{\mathrm{T}}$，其中 $P(a_j)$为产生字元符号 a_j 的概率。假设有 k 个符号源产生，则由大数法则可知，若 k 够大，a_j 平均会输出 $kP(a_j)$次，因此由 k 个输出所得的平均自信息量为

$$-\sum_{j=1}^{J}kP(a_j)\log P(a_j) \tag{7.3-2}$$

于是观察每个信息源输出所得的平均信息量 $H(\mathbf{z})$为

$$H(\mathbf{z})=-\sum_{j=1}^{J}P(a_j)\log P(a_j) \tag{7.3-3}$$

$H(\mathbf{z})$ 称为信息源的不确定性 (uncertainty) 或熵 (entropy)。

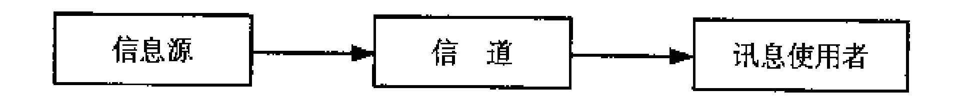


图 7.3-1 一个简单信息系统的示意图

接着看信道的影响。设信道输出的可能符号集为 $B=\{b_1,b_2,\cdots,b_K\}$，其对应的概率集为 $\mathbf{v}=[P(b_1), P(b_2),\cdots, P(b_k)]^{\mathrm{T}}$，其中 $P(b_k)$为产生符号 b_k 的概率。由完全 (total) 概率可知

$$P(b_k)=\sum_{j=1}^{J}P(b_k \mid a_j)P(a_j) \tag{7.3-4}$$

其中 $P(b_k \mid a_j)$ 代表条件概率。为方便起见可将所有的条件概率以信道 (转移) 矩阵来表示

$$\boldsymbol{Q}=\begin{bmatrix} P(b_1 \mid a_1) & P(b_1 \mid a_2) & \cdots & P(b_1 \mid a_J) \\ P(b_2 \mid a_1) & P(b_2 \mid a_2) & \cdots & P(b_2 \mid a_J) \\ \vdots & \vdots & \vdots & \vdots \\ P(b_K \mid a_1) & P(b_K \mid a_2) & \cdots & P(b_K \mid a_J) \end{bmatrix} \tag{7.3-5}$$

因此整个输出符号的概率分布为

$$\boldsymbol{v}=\boldsymbol{Q}\boldsymbol{z} \tag{7.3-6}$$

接下来我们要知道具有信道矩阵 $\boldsymbol{Q}$ 的信道的容量 (capacity)。首先我们计算在信息使用者观察得某一个输出 b_k 的情况下，信息源的熵，即

$$H(\boldsymbol{z} \mid b_k)=-\sum_{j=1}^{J} P(a_j \mid b_k)\log P(a_j \mid b_k) \tag{7.3-7}$$

对上式中所有 b_k 取期望值或平均得

$$H(\boldsymbol{z} \mid \boldsymbol{v})=\sum_{k=1}^{K} H(\boldsymbol{z} \mid b_k)P(b_k) \tag{7.3-8}$$

上式经简化后得

$$H(\boldsymbol{z} \mid \boldsymbol{v})=-\sum_{j=1}^{J}\sum_{k=1}^{K} P(a_j,b_k)\log P(a_j \mid b_k) \tag{7.3-9}$$

其中 $P(a_j, b_k)$ 为 a_j 与 b_k 的联合概率。其意义为在观察得一个输出符号下，一个信息源符号的平均信息量。因为 $H(\boldsymbol{z})$ 是在未观察得输出符号前--信息符号的平均信息量，所以 $H(\boldsymbol{z})$ 与 $H(\boldsymbol{z} \mid \boldsymbol{v})$ 之差代表观察得一个输出符号所获得的平均信息量。此差量称为 $\boldsymbol{z}$ 与 $\boldsymbol{v}$ 的互 (mutual) 信息量，表示成 $I(\boldsymbol{z},\boldsymbol{v})$

$$I(\boldsymbol{z},\boldsymbol{v})=H(\boldsymbol{z})-H(\boldsymbol{z} \mid \boldsymbol{v}) \tag{7.3-10}$$

将 $H(\boldsymbol{z})$ 与 $H(\boldsymbol{z} \mid \boldsymbol{v})$ 以 (7.3-3) 式及 (7.3-9) 式代入，再利用 $P(a_j)=P(a_j, b_1)+P(a_j, b_2)+\cdots+P(a_j, b_K)$ 的事实可得

$$I(\boldsymbol{z},\boldsymbol{v})=\sum_{j=1}^{J}\sum_{k=1}^{K} P(a_j,b_k)\log\frac{P(a_j,b_k)}{P(a_j)P(b_k)} \tag{7.3-11}$$

上式可进一步简化成

$$I(\boldsymbol{z},\boldsymbol{v})=\sum_{j=1}^{J}\sum_{k=1}^{K} P(a_j)q_{kj}\log\frac{q_{kj}}{\sum_{i=1}^{J} P(a_i)q_{ki}} \tag{7.3-12}$$

其中 $q_{kj}=P(b_k|a_j)$。换言之，观察得信道的一个输出所收到的平均信息量与信源的概率分布 $\boldsymbol{z}$ 及信道矩阵有关。当输入与输出符号为统计无关时，其 $I(\boldsymbol{z},\boldsymbol{v})=0$ 为最小值。因为此时 $P(a_j,b_k)=P(a_j)P(b_k)$。在所有可能的概率分布 $\boldsymbol{z}$ 中 $I(\boldsymbol{z},\boldsymbol{v})$ 的最大值即是通道矩阵 $\boldsymbol{Q}$ 所呈现的信道容量 C。亦即

$$C=\max_{z}[I(\boldsymbol{z},\boldsymbol{v})] \tag{7.3-13}$$

此信道容量定义出可在此信道可靠传送的最大速率。此外，信道容量与信息源的输入概率无关，只与定义信道的条件概率有关。

我们举一个例子来说明以上所提到的观念，考虑一个二元信息源，其符号集 $A=\{a_1,a_2\}=\{0,1\}$，其中对应的概率分别为 $P(a_1)=p$ 及 $P(a_2)=1-p=\bar{p}$。由 (7.3-3) 式可得此信息源的熵为

$$H(\boldsymbol{z})=-p\log_2 p-\bar{p}\log_2 \bar{p}$$

此式只与 p 有关且上式等号右边称为二元熵函数 (binary entropy function) 表示成 $H_b(\cdot)$。图 7.3-2 (a) 显示此函数。图中显示当 $p=1/2$ 时，此函数有最大值 (1 位)。

设传输信道是错误概率为 p_e 的二元对称信道 (binary symmetric channel, BSC)，其信道矩阵为

$$\boldsymbol{Q}=\begin{bmatrix}1-p_e & p_e\\ p_e & 1-p_e\end{bmatrix}=\begin{bmatrix}\bar{p}_e & p_e\\ p_e & \bar{p}_e\end{bmatrix}$$

对一个二元输入，BSC 的输出符号集 $B=\{b_1,b_2\}=\{0,1\}$。由 (7.3-6) 式知其对应的概率为

$$\boldsymbol{v}=\boldsymbol{Q}\boldsymbol{z}=\begin{bmatrix}\bar{p}_e & p_e\\ p_e & \bar{p}_e\end{bmatrix}\begin{bmatrix}p\\ \bar{p}\end{bmatrix}=\begin{bmatrix}\bar{p}_e p+p_e\bar{p}\\ p_e p+\bar{p}_e\bar{p}\end{bmatrix}=\begin{bmatrix}p(b_1)\\ p(b_2)\end{bmatrix}$$

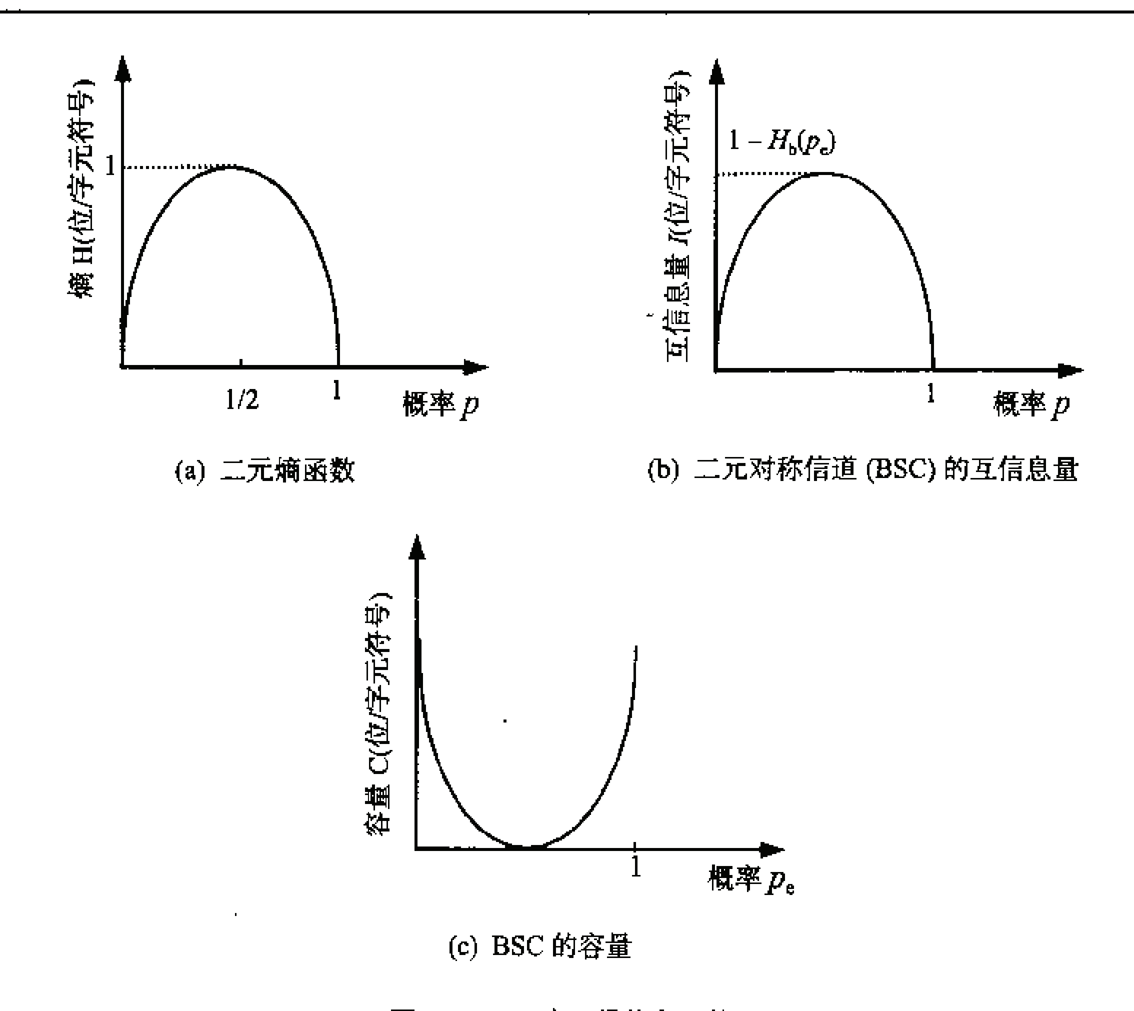
现在由 (7.3-12) 式计算 BSC 的互信息量可得

$$I(\boldsymbol{z},\boldsymbol{v})=H_b(pp_e+\bar{p}\ \bar{p}_e)-H_b(p_e)$$

图 7.3-2 (b) 显示在某个固定的 p_e 时，$I(\boldsymbol{z},\boldsymbol{v})$ 对 p 的曲线图形。由图中可看出，当 $p=0$ 或 1 时 $I(\boldsymbol{z},\boldsymbol{v})=0$，此与直觉相符，因为永远固定只传送 0 或 1，所以收到任何一位时没有因而增加任何新信息，即 $H(\boldsymbol{z})=H(\boldsymbol{z}|\boldsymbol{v})$，因此 $I(\boldsymbol{z},\boldsymbol{v})=0$。$I(\boldsymbol{z},\boldsymbol{v})$ 的最大值发生在两位有同等概率时，此时所得的新数据具有最不确定性，因而收到最大的信息量。由 (7.3-13) 式及图 7.3-2 (b) 中对所有概率分布 $\boldsymbol{z}=[p,1-p]$ 所得到的 $I(\boldsymbol{z},\boldsymbol{v})$ 图中可知

$$C=1-H_b(p_e)$$

如图 7.3-2 (c) 所示。当 $p_e=0$ 或 1 时，C 最大 (1 位／字元符号)，此时信道的输出完全可以预期。但是当 $p_e=1/2$ 时，信道的输出完全不可预期，因此任何信息都无法通过此信道传输。



(a) 二元熵函数

(b) 二元对称信道 (BSC) 的互信息量

(c) BSC 的容量

图 7.3-2　三个二元信息函数

7.3-3　基本编码定理

本节将介绍三个有关编码的基本定理：(1) *无噪声* (noiseless)下的编码定理；(2) *有噪声* (noisy)下的编码定理；(3) *信源* (source) 编码定理。无噪声影响使得一个通讯系统中数据传送可以没有任何错误，因此无噪声下的编码的重点在于如何以最短的平均码长很精简地来表示每个字元符号。有噪声影响使得一个通讯系统中数据传送容易发生错误，因此有噪声下编码的重点在于如何提供可靠的通讯，减少错误发生。信源编码的精髓是假设在无噪声信道中，在某个失真范围内如何以最小的位率将信息传送给使用者。换言之，这是一个讨论位率与失真程度间权衡取舍的一个定理，也是*率失真* (rate-distortion) 理论中非常重要的结果。

- **无噪声下的编码定理**

此定理又称*仙农* (Shannon) 的第一定理。考虑一无记忆信源，亦即其字元符号为统计独立。设 $A=\{a_1,a_2,\cdots,a_J\}$仍为可能的字元符号集，其对应的概率仍为 $P(a_1),P(a_2),\cdots,P(a_J)$，若以 n 个字元符号为一个输出单元，则总共可能有 J^n 个可能的值，令此值的集合为 $A'=\{\alpha_1,\alpha_2,\cdots,\alpha_{J^n}\}$，其中每个$\alpha_i$是由 A 中的 n 个字元符号所组成。设 $P(\alpha_i)$ 为伴随α_i的概率，因此

$$P(\alpha_i)=P(a_{j1})P(a_{j2})\cdots P(a_{jn}) \tag{7.3-14}$$

其中 a_j 的额外下标是指明形成 α_i 的 n 个字元符号。设 $\mathbf{z}'=\{P(\alpha_1),P(\alpha_2),\cdots,P(\alpha_{J^n})\}$，则其熵为

$$H(\mathbf{z}')=-\sum_{i=1}^{J^n}P(\alpha_i)\log P(\alpha_i)$$

将 (7.3-14) 式代入上式中并化简后可得

$$H(\mathbf{z}')=nH(\mathbf{z}) \tag{7.3-15}$$

因 α_i 的自信息为 $\log\frac{1}{P(\alpha_i)}$，因此一个用来代表 α_i 的码字的合理整数长度 $l(\alpha_i)$ 应为

$$\log\frac{1}{P(\alpha_i)}\leqslant l(\alpha_i)\leqslant\log\frac{1}{P(\alpha_i)}+1 \tag{7.3-16}$$

上式乘上 $P(\alpha_i)$ 并对所有 i 求总和可得

$$\sum_{i=1}^{J^n}P(\alpha_i)\log\frac{1}{P(\alpha_i)}\leqslant\sum_{i=1}^{J^n}P(\alpha_i)l(\alpha_i)<\sum_{i=1}^{J^n}P(\alpha_i)\log\frac{1}{P(\alpha_i)}+1$$

或可写成

$$H(\mathbf{z}')\leqslant L'_{\text{avg}}<H(\mathbf{z}')+1 \tag{7.3-17}$$

其中 L'_{avg} 代表码字的平均长度。将上式除以 n 并利用 (7.3-15) 式可得

$$H(\mathbf{z})\leqslant\frac{L'_{\text{avg}}}{n}<H(\mathbf{z})+\frac{1}{n} \tag{7.3-18}$$

取其极限变成

$$\lim_{n\to\infty}\left[\frac{L'_{\text{avg}}}{n}\right]=H(\mathbf{z}) \tag{7.3-19}$$

(7.3-19) 式是说如果我们考虑一次取无限长字元符号来编码，理论上此编码结果可任意逼近 $H(\mathbf{z})$，而这正是 L'_{avg}/n 的下界、因此编码效率 (efficiency) η 可定义成

$$\eta=\frac{H(\mathbf{z})}{\frac{L'_{\text{avg}}}{n}}\quad 或\quad \eta=n\frac{H(\mathbf{z})}{L'_{\text{avg}}} \tag{7.3-20}$$

- **有噪声下的编码定理**

此定理又称仙农的第二定理。在有噪声信道的情况下，我们较关切的是信息传送可以多

么可靠？此定理是说，已知一离散噪声无记忆信道的容量为 C，且有一正速率为 R 的信源，其中 $R < C$，则存在有一种编码，使信源输出在此信道传输时，具有任意小的错误概率。此定理预测在有噪声存在时，可以称得上是无错误的传输。不过此定理只告诉我们有这种码的存在，并未告诉我们如何构造这些码。这个问题属于信道编码 (channel coding) 或错误更正码 (error correcting code) 的领域，已超过本书的范围，故不再继续讨论。有兴趣的读者可以在图书馆找到专门讨论本问题的参考书籍。

● **信源编码定理**

先前曾以信道矩阵 $\boldsymbol{Q}$ 代表信源输出至接收端产生错误状况的一种模型。假设信道传输不再有任何错误的可能，而接收端解码所得与信源输出之间的唯一误差是来自数据压缩与解压缩间的失真。此时可沿用矩阵 $\boldsymbol{Q}$ 的概念，将其当成此失真情况的另一个模型。此时矩阵中一元素 q_{kj} 代表一个信源符号 a_j，经编码与解码的过程得到输出符号 b_k 的概率。

令 $\rho(a_j, b_k)$ 代表上述失真度。对所有符号的平均失真 $d(\boldsymbol{Q})$ 可写成

$$d(\boldsymbol{Q}) = \sum_{j=1}^{J}\sum_{k=1}^{K}\rho(a_j, b_k)P(a_j, b_k)$$
$$= \sum_{j=1}^{J}\sum_{k=1}^{K}\rho(a_j, b_k)P(a_j)q_{kj} \tag{7.3-21}$$

设 $d(\boldsymbol{Q})$小于或等于 D 的所有编码解码过程的转移概率集合为

$$\boldsymbol{Q}_D = \{q_{kj} \mid d(\boldsymbol{Q}) \leqslant D\} \tag{7.3-22}$$

因为每一个编码解码过程是由一个假想的信道矩阵 $\boldsymbol{Q}$ 所定义，所以观察一个解码器输出所获得的信息可参照 (7.3-12) 式来计算。因此我们可定义出一个失真函数

$$R(D) = \min_{Q \in Q_D}[I(\boldsymbol{z}, \boldsymbol{v})] \tag{7.3-23}$$

上式定义了在平均失真小于或等于 D 的条件之下，信源可以传给使用者的最小码率。要计算 $R(D)$，可在下列条件限制下选取适当的 $\boldsymbol{Q}$ 使 $I(\boldsymbol{z},\boldsymbol{v})$最小化：

$$q_{kj} \geqslant 0 \tag{7.3-24}$$

$$\sum_{k=1}^{K} q_{kj} = 1 \tag{7.3-25}$$

以及

$$d(\boldsymbol{Q}) = D \tag{7.3-26}$$

其中 (7.3-24) 与 (7.3-25) 式是由信道矩阵的基本性质而来，(7.3-26) 式则表示最小的信息失真码率出现在最大可能失真时。

一个典型的率失真函数如图 7.3-3 所示。通常只有对于简单的信源以及简单的失真测度

才有可能以解析的方式获得率失真函数，因此大多需要有适当的迭代算法在计算机上计算出所需要的 $R(D)$函数。

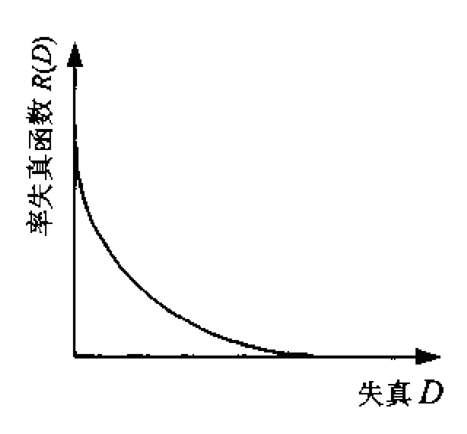


图 7.3-3 一个典型的率失真函数图形

7.4 无失真压缩

在某些数字信号应用中，无失真压缩是较能被接受的减少数据的方法。例如许多医学或公务文件的图像，由于法律的因素，有损压缩一般而言并不适用；另一种应用为大地卫星图像的处理，无论是应用还是数据采集的代价都较高，因而不希望信息有损失；还有一种是数字 X 光摄像，在此处任何信息的损失都可能会影响诊断的正确性。本节中将针对目前通用的无失真压缩方法进行讨论。

● **变长编码**

假设有一原始符号序列为

$$a_1a_4a_4a_1a_2a_1a_1a_4a_1a_3a_2a_1a_4a_1a_1$$

其中含四个符号元素 a_1、a_2、a_3、a_4，故可编码为

a_1	00
a_2	01
a_3	10
a_4	11

于是我们可以把该符号序列储存为

a_1	a_4	a_4	a_1	a_2	a_1	a_1	a_4	a_1	a_3	a_2	a_1	a_4	a_1	a_1
00	11	11	00	01	00	00	11	00	10	01	00	11	00	00

总共占 30 位。不过我们观察 a_1、a_2、a_3、a_4 出现的频率并不一样，其实际出现次数统计如下：a_1 有 8 次，a_2 有 2 次，a_3 有 1 次，a_4 有 4 次。于是我们把编码的方式改为

a_1	0
a_2	110
a_3	111
a_4	10

其中越常出现的符号元素编码就以越短的码来表示，如此可重新编码成

a_1	a_4	a_4	a_1	a_2	a_1	a_1	a_4	a_1	a_3	a_2	a_1	a_4	a_1	a_1
0	10	10	0	110	0	0	10	0	111	110	0	10	0	0

只需用 25 位来表示，这就是变长编码的基本构想。

● **霍夫曼编码**

霍夫曼 (Huffman) 编码是由 D. A. Huffman 于 1952 年提出，当时发表在一篇名为 "A Method for the Construction of Minium Redundancy Codes" 的论文上。当对信源符号一次一个个别编码时，霍夫曼编码对每个原始符号所产生的码具有最短的平均码长，因此是最佳的编徨

霍夫曼法首先根据所考虑的原始符号的概率大小顺序产生一个原始码序列，接着合并两个最小概率的符号为一个新符号并加入下一次缩减。图 7.4-1 显示一霍夫曼编码的过程。在图的最左边，一个虚构的原始符号及其概率集合，从上到下依照概率的递减顺序排列。为了构成第一次原始码的缩减，底部的两个概率 0.06 与 0.04 被合并成一个复合符号，其概率为 0.1。该符号被置于第一次原始码缩减列并仍按递减顺序排列，这样的过程一直重复到缩减的原始码剩下两个符号为止。

原来信源		信源缩减			
符号	概率	1	2	3	4
a_2	0.4	0.4	0.4	0.4	0.6
a_6	0.3	0.3	0.3	0.3	0.4
a_1	0.1	0.1	0.2	0.3	
a_4	0.1	0.1	0.1		
a_3	0.06	0.1			
a_5	0.04	0.1			

图 7.4-1　霍夫曼码的设计过程

最后，从剩下的两个符号码开始回溯到最初的原始码进行编码。此时分配给两个符号最短长度的代码，即 0 与 1，如图 7.4-2 所示 (或分配成 1 与 0)。由于概率 0.6 的缩减符号是由它左边的两个缩减符号合并的，所以它的代码 0 也应属于前面两个符号，同时也附加上 0 与 1 (或 1 与 0 但需与前面一致)，这样的步骤一直重复地对所有缩减符号进行，直到最初的符号为止。最后产生的码字显示于图 7.4-2 的左方。该编码的平均长度为

$$
\begin{aligned}
L_{avg} &= (0.4)(1) + (0.3)(2) + (0.1)(3) + (0.1)(4) + (0.06)(5) + (0.04)(5) \\
&= 2.2 \quad (\text{位/码元})
\end{aligned}
\tag{7.4-1}
$$

该信源的熵为 2.14 位／符号，故霍夫曼码的效率为 $2.14/2.2 = 0.973$。

原来信源			信源缩减							
符号	概率	代码	1		2		3		4	
a_2	0.4	1	0.4	1	0.4	1	0.4	1	0.6	0
a_6	0.3	00	0.3	00	0.3	00	0.3	00	0.4	1
a_1	0.1	011	0.1	011	0.2	010	0.3	01		
a_4	0.1	0100	0.1	0100	0.1	011				
a_3	0.06	01010	0.1	0101						
a_5	0.04	01011								

图 7.4-2　霍夫曼码的分配过程

霍夫曼编码过程为一组符号按概率产生最佳编码，但须满足一次将一个符号编码的条件限制。在代码码字产生后，解码时只需借助简单的查表便可完成。该代码本身是一个*即时惟一可解* (instaneous and uniquely decodable) 的区块码。称为区块码是因为每一个原始符号被映射为一固定代码符号序列；称为即时是因为不需参考后续代码符号便可以将代码符号字串中的代码码字解码；称为惟一可解是因为任何代码的码字串都只有一种解码结果。

● **霍夫曼编码算法的过程**

霍夫曼编码算法必须扫描原始图像数据两次，其步骤如下：

(1) 读入数据并统计原始数据中各个数据出现的次数，当做出现频率。

(2) 建立霍夫曼树。接着把霍夫曼树转成码表再次读入图像文件的内容，依码表转成码。

经过上一步骤编码的过程后，其结果有两种特性：

1. 出现率越高的数据元素有较短的码，反之则有较长的码。
2. 编码的结果具有惟一的特色，即有*惟一前置码* (unique prefix)。

整个编码的详细实现过程说明如下：

步骤一：

首先读入整个图像文件，以统计每个图像数据出现的次数，以作为构造霍夫曼树的依据。

步骤二：

构造霍夫曼树。霍夫曼树的一般型态如图 7.4-3 所示。从图 7.4-3 中可以看出，方块为末端节点，代表原始图像文件中曾出现过的数据，圆圈则为内部节点。每个*分支* (branch) 都被编以 0 或 1 的码，因此阶层数越少的话，该层的节点被编出的码就越短，例如：阶层 n 的节点 A 及 B 的码长均为 n，位于阶层 $n-1$ 的节点 C 的码长为 $n-1$，位于阶层 1 的节点 X 的码长为 1。建立霍夫曼树是由底层向上层建立，霍夫曼树底层的节点有最长的码，越往上层的节点，它的码越短。所以一开始必须找出概率较小的数据当成节点放入霍夫曼树中。建立霍夫曼树可以分为以下几个子步骤来形成：

子步骤 1：将原始图像数据内容中每个出现过的数据元素当成末端节点，形成节点集合，并将每个节点的出现概率当成该节点的权重。

子步骤 2：从节点集合中找出权重最小的两个节点 X 与 Y。

子步骤 3：将 X 与 Y 的权重相加，作为新节点 W 的权重，并在节点集合中移除 X 与 Y 节点，加入 W 节点。

子步骤 4：重复 2 与 3 的子步骤直到节点集合只剩下一个节点为止 (此节点为根节点)。

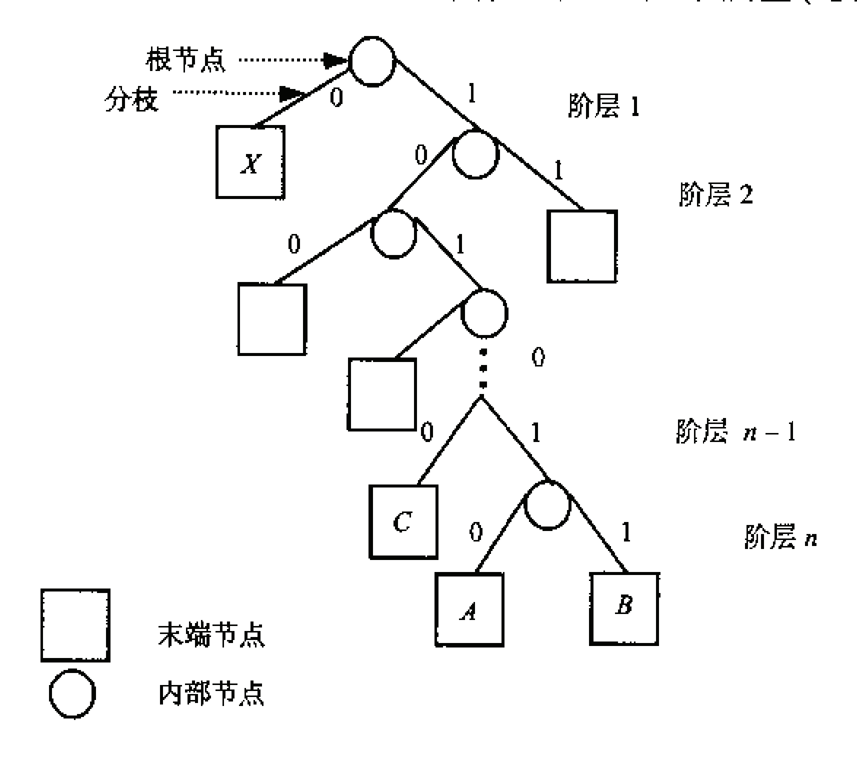


图 7.4-3　霍夫曼树的一般型态

步骤三：

建立码表，如图 7.4 - 4 所示。假设有 $a_1,a_2,\cdots,a_6$ 共六个 8 位的像素，且其出现的次数分别为 10,40,6,10,4 及 30，共计 100 次。

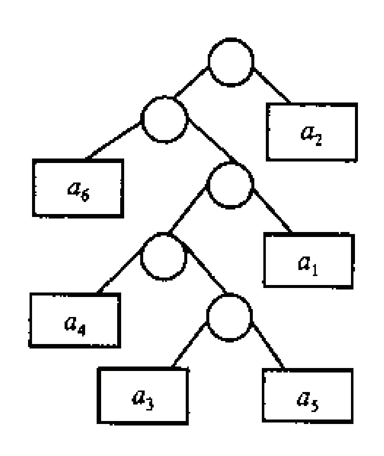


图 7.4-4　码　表

接下来把霍夫曼树转成码表。先把图中每一个末端节点改写成对应的数据，每个非末端

的节点有左右两个节点分枝。编码的原则为：从根节点开始，向左填 0，向右填 1，一直填到末端节点为止，这个操作称为树的追踪。每个数据的码就是根节点到末端节点的分枝上的码，如图 7.4-5 所示。从图中可以看出，出现率高的编以较短的码，出现率低的则编较长的码。整个码表只有两栏：数据与码，供给下一步骤查表。

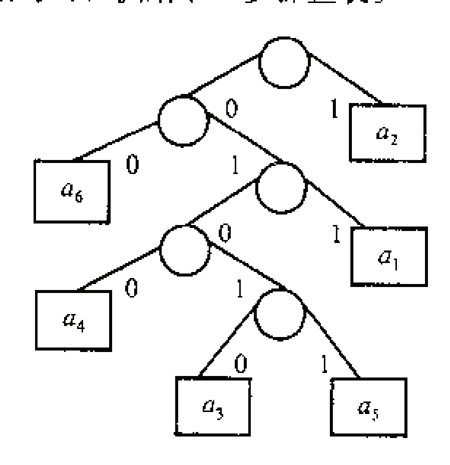


图 7.4-5 树的追踪

步骤四：

此步骤为查表写入代码。首先重新读入原始数据内容，依读入数据查出其在码表中对应的码，写至输出文件，接着重复查表与写入的操作，直到读入 EOF 为止。依步骤三所得的图表所示的码表将原始的数据编码，编码后数据的长度为 $3\times10 + 1\times40 + 5\times6 + 4\times10 + 5\times4 + 2\times30 = 220$ 位。原本 a_1 至 a_6 各要用 8 位表示，故共需要 $8\times100 = 800$ 位，由此可见所节省的位数相当可观。

● 游程长度编码

游程长度编码 (run length encoding, RLE) 是一种既简单又快速的压缩方法，特别适合用在如小画家之类的绘图软件所产生的图像文件，通常会有 2～3 倍的压缩率，因此许多绘图软件都采用此种压缩方法。此外，RLE 也是传真机压缩编码标准所采用的概念。

游程长度编码的概念是将一连串重复的数据用两个字节 (byte) 表示，第一个字节代表该字串的长度 (重复次数)，第二个字节才是数据。如表 7.4-1 所示，原始数据为 17 个字节，经压缩编码成 12 个字节，故省掉了 5 个字节。

表 7.4-1 游程长度编码的例子

原始数据	$a_1\,a_1\,a_1\,a_1\,a_1$	$a_2\,a_2$	a_3	$a_4\,a_4\,a_4$	$a_5\,a_5\,a_5\,a_5$	$a_6\,a_6$
编　码	$5a_1$	$2a_2$	$1a_3$	$3a_4$	$4a_5$	$2a_6$

由上面的例子中可明显看出只有当重复的次数大于 2 时，才会有数据压缩的效果。若重复次数恰好等于 2，则没有数据压缩的效果，但也没有膨胀。比较不好的状况是，重复的次数为 1 (亦即没有重复)，则压缩后反而会比原始的数据多出一个字节，亦即有膨胀的现象。在最坏的情况下，原始数据中没有任何相邻的两个字节的值相同，此时以游程长度编码压缩

后的结果会使数据量膨胀成原来的两倍大！这是游程长度编码最大的缺点，因此有人提出几个改进的方法。

● PCX 文件的压缩法

PCX 文件是 PC PaintBrush 软件所产生的图形文件，其图形压缩的方法就称为 PCX 的 RLE，基本上它是由上述游程长度编码的概念改进而来的。其改进重点在于编码时，若碰到重复次数为 1 的数据，则原封不动输出该字节，而当重复次数大于 1 时，则以两个字节来表示，其中前一个字节为重复次数加上 c0h (也就是 bit7 与 bit6 均设为 1)，后一个字节才是数据，如表 7.4-2 所示。由表中可知编码后只用掉 11 个字节，比先前基本的 RLE 又省了一个字节。以上为压缩的编码过程，解压缩的解码过程则可参考图 7.4-6 的流程图。

表 7.4-2　PCX 的 RLE 压缩法

原始数据	$a_1\,a_1\,a_1\,a_1\,a_1$	$a_2\,a_2$	a_3	$a_4\,a_4\,a_4$	$a_5\,a_5\,a_5\,a_5$	$a_6\,a_6$
编　码	$\underline{c5}a_1$	$\underline{c2}a_2$	a_3	$\underline{c3}a_4$	$\underline{c4}a_5$	$\underline{c2}a_6$

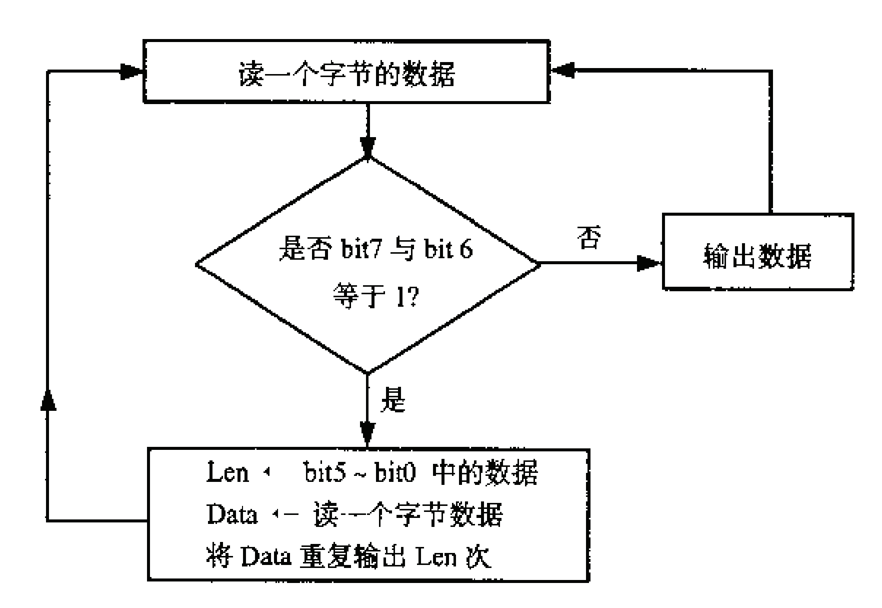


图 7.4-6　PCX 的 RLE 解压缩法

因为 PCX 的 RLE 使用 6 位来表示重复的次数，而 6 位所能表示的最大值为 63，因此若重复次数超过 63，则该笔数据就必须用 4 个字节或更多的字节来表示，例如有 128 个连续的 a，则要用 6 个字节表示成 $\underline{ff}\,a\,\underline{ff}\,a\,\underline{c2}\,a$。

另外，还有一种要考虑的情况是当重复次数为 1，且该数据的值又大于 c0h 时。此时必需输出两个字节，前一个为 c1h (表示重复一次)，第二个才是数据本身，这是 PCX 的 RLE 的缺点。

● CUT 文件的压缩法

Dr. Halo 绘图软件也是采用以 RLE 为基础的方法来做压缩，因其图形文件以 CUT 为后缀，所以也称为 CUT 文件。表 7.4-3 显示一个 CUT 文件压缩的例子。

表 7.4-3　CUT 文件压缩的例子

原始数据	$a_1\ a_1\ a_1\ a_1\ a_1$	$a_2\ a_3\ a_4$	$a_5\ a_5$	$a_1 a_2\ a_4\ a_6$
编　码	$\underline{85}a_1$	$\underline{03}a_2\ a_3\ a_4$	$\underline{82}a_5$	$\underline{04}\ a_1 a_2\ a_4\ a_6$

此处连续出现的数据仍用两个字节来表示，前一个代表重复次数加上 80h (亦即 bit7 设为 1)，后一个字节才是数据。对于不重复的数据将其收集成串，并在最前面加上一个字节，此字节记录其后共有多少个字节是不重复的数据 。整个 CUT 文件解压缩的程序如图 7.4-7 所示。

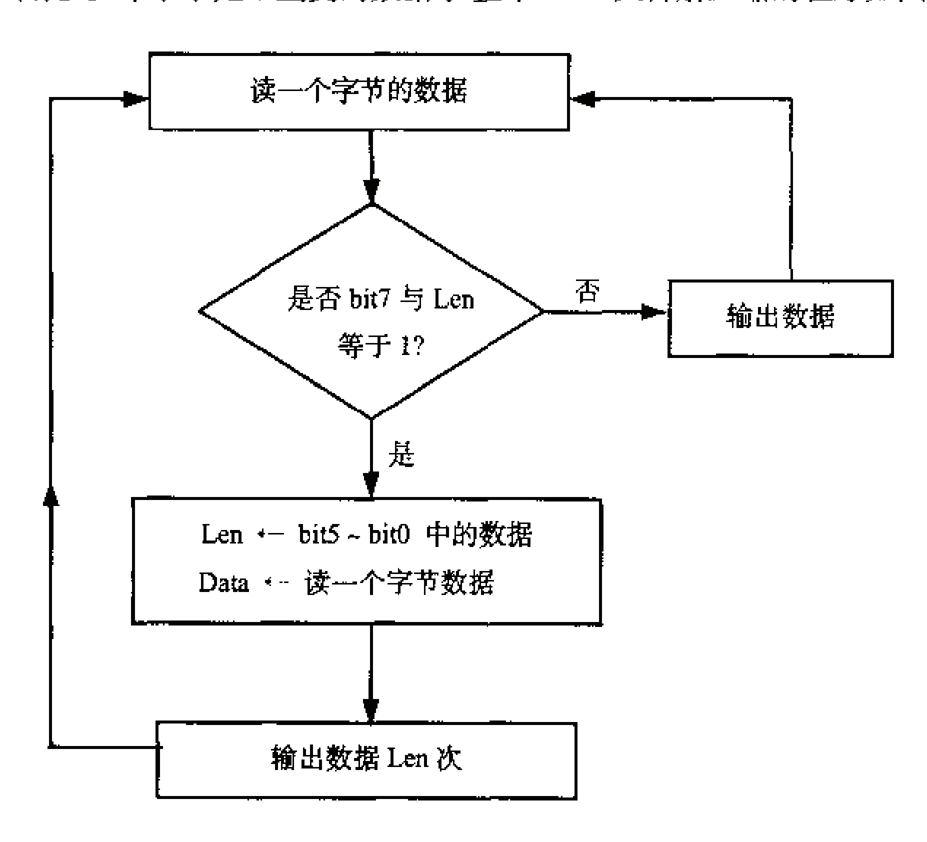


图 7.4-7　CUT 文件的解压缩流程

CUT 文件的 RLE 较 PCX 文件的 RLE 为优，这是因为前者只使用一位来辨别是否重复，而后者要用两位。

7.5　有损压缩

所谓有损 (lossy) 压缩是指带有失真的压缩，在信息理论中也称为熵压缩 (entropy compression)。考虑有损压缩的一个简单例子：在监测取样值时只有当取样值超过某个预设的临界值时，才传送数据。如果这样的情况愈不常出现，则可以实现信号空间的压缩也会愈大，但在此同时，实际的原始采样值就无法精确地恢复，亦即有信息损耗。以下是有损压缩的基本特性：

1. 有冗余性就可以压缩。

2. 压缩只能在一定限度之内可逆 (恢复)。
3. 超过此限度，必然带来失真。
4. 允许的失真越大，压缩的比例也可以越大。

实现有损压缩主要有两大方法：特征提取和量化方法，如图 7.5-1 所示。特征提取的典型例子如指纹的模式识别：一旦提取出足以有效代表和区分不同人指纹的有效参数，便可用其取代原始指纹数据。有关特征提取部分我们将在第十章加以讨论。对于实际压缩技术的应用而言，量化是更为通用的有损压缩技术。除了对无记忆 (memoryless) 信号的单一样本做所谓的无记忆量化外，也可以将有记忆信号的多个相关样本映射到不同空间，去除原始数据中的相关性后再做量化处理。本章在后面的讨论中均是以这方面为重点，并分别以国际图像压缩标准 JPEG 与小波变换作为实例说明。

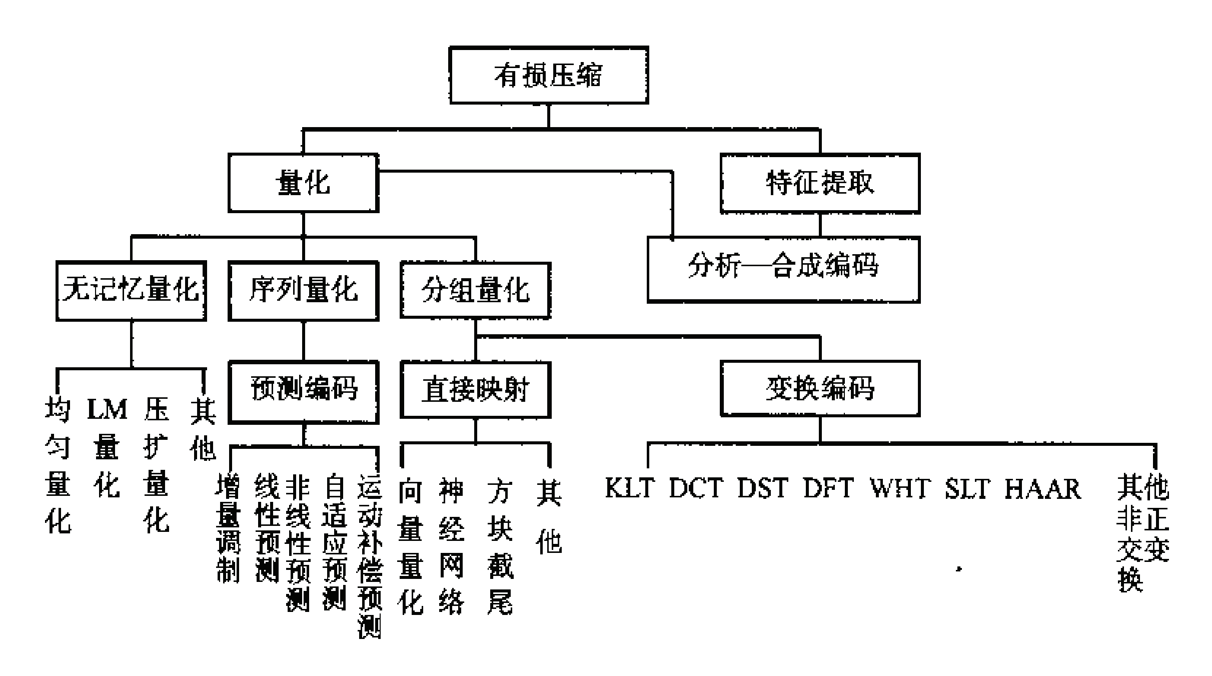


图 7.5-1　有损压缩技术的简单分类

7.6 图像压缩标准 JPEG

JPEG 是目前国际标准组织所订的图像压缩标准，其特性如下：

1. JPEG 基本上是一种有损压缩技术，不过可以做到有限度的失真而使眼睛辨认不出来，但却能达到极大的压缩量。
2. JPEG 所处理的图像内容并不限定要何种型态，从照片到卡通图画，甚至是文字的扫描图像都能够处理，但对于较自然的图像内容如照片，会有最好的效果。至于图像的色彩也没有限制，一般灰度图像及全彩的图像都能处理。
3. 使用者可以在图像的品质与压缩比间选一折衷点。由于图像的品质和压缩比是相对的，压缩比越大品质自然越差。有些图像是供浏览用的，品质较差也无所谓，因此可以指定较高的压缩比。如果要求较好的品质，可以指定较低的压缩比。

4. JPEG 提供四种压缩方式。最常用的是循序模式，图像由左至右，由上至下循序处理。第二种是级进模式，当图像数据处理很慢时，例如通过网络传送，可以先传一个模糊图像，再渐渐使图像清晰。第三种是以不失真的方式压缩图像，但这种方式很少被使用。第四种则是阶层式的压缩，将图像以多重解析度来压缩，使其可以被用在较低解析度下而不需要将整个图像解压缩。

接下来我们探讨 JPEG 的循序模式的压缩原理。

- **JPEG 的压缩与解压缩过程**

图 7.6-1 是 JPEG 的压缩过程。在压缩时是先把图像分割成 8×8 的块来处理，不足的部分则以最右边或最前面的点延伸为 8 的倍数。解压缩的过程只是压缩的逆向过程，因此，此处只着重压缩部分。

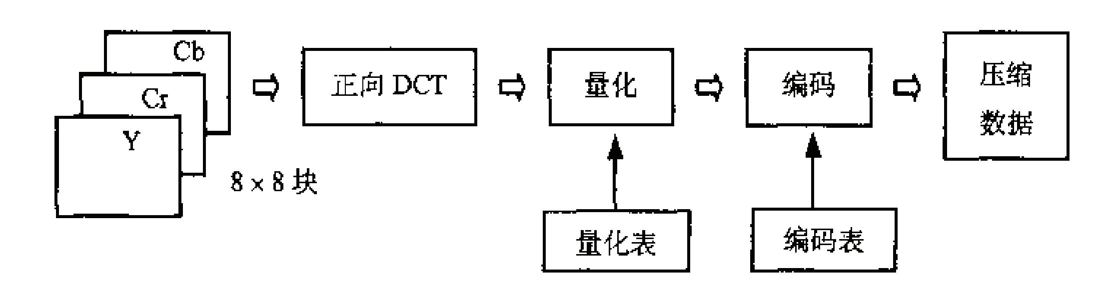


图 7.6-1 JPEG 的压缩过程

- **离散余弦变换**

第四章中所提到的离散余弦变换 (discret cosine transform, DCT)，是整个 JPEG 压缩技术的精髓所在。其一维的定义为

$$F(k)=\frac{2c(k)}{N}\sum_{n=0}^{N-1}f(n)\cos\left[\frac{(2n+1)k\pi}{2N}\right],\quad c(k)=\begin{cases}\frac{1}{\sqrt{2}}, & k=0\\ 1, & k\neq 0\end{cases} \tag{7.6-1}$$

其中 $f(n)$ 为输入数据、$F(k)$ 为变换的输出、$0\leqslant n,\ k\leqslant N-1$。

我们举例说明 DCT 的效应。在表 7.6-1 中有三组数据，每组各有 8 个数据 ($N=8$) 及其经过 DCT 变换后各频率的系数。在第一组的数据中，由于数据间没有变化，因此变换后只有直流系数 DC，其他的交流系数 AC 全为 0。第二组数据间的变化很小、振幅不大，因此 AC 值也都很小。第三组的数据则因为变动比较大，故有较大的 AC 值。

表 7.6-1 DCT 效应的例子

$f(n)$	6	6	6	6	6	6	6	6
$F(k)$	16.97	0.0	0.0	0.0	0.0	0.0	0.0	0.0
$f(n)$	6.1	6.4	5.6	6.2	5.2	5.8	6.0	6.5
$F(k)$	16.90	0.01	0.75	-0.39	0.07	0.09	-0.23	-0.72
$f(n)$	1.2	3.4	1.0	2.0	5.0	2.1	4.0	8.0
$F(k)$	9.44	-4.18	1.84	-1.40	2.02	-2.95	-1.57	0.52

DCT 在图像压缩中可扮演何种角色？首先观察一些具有所谓连续色调 (continuous tone) 的自然图像 (例如照片)，我们将会发现图像上大多数局部区域内的颜色或灰度变化都不大。由表 7.6-1 的例子可知，若以 DCT 对这些图像数据分析，就会发现较高频的 AC 值都很小。由于人的眼睛对低频部分比对高频的部分要敏感些，因此若去除高频的部分，则在图像还原时，只产生不易辨认出来的些许误差。换言之，去掉视觉上较不重要的高频部分，对图像品质影响不大，却能提高压缩比。

JPEG 使用二维的第二类 DCT，处理一个 8×8 的二维矩阵数据，因此变换成为 64 个系数 (1 个 DC 与 63 个 AC)，称为顺向的 DCT (DCT 正变换) (forward DCT 或 FDCT)；逆向的 DCT (DCT 反变换) (inverse DCT 或 IDCT) 则是将此 64 个系数还原为 8×8 的数据，用于解压缩时。其公式如下：

FDCT

$$F(k,l)=\frac{c(k)c(l)}{4}\sum_{m=0}^{7}\sum_{n=0}^{7}f(m,n)\cos\frac{(2m+1)k\pi}{16}\cos\frac{(2n+1)l\pi}{16} \tag{7.6-2}$$

IDCT

$$f(m,n)=\frac{1}{4}\sum_{k=0}^{7}\sum_{l=0}^{7}c(k)c(l)F(k,l)\cos\frac{(2m+1)k\pi}{16}\cos\frac{(2n+1)l\pi}{16} \tag{7.6-3}$$

FDCT 产生 64 个系数后，接着对其扫描，其扫描顺序则如图 7.6-2 所示。由于其扫描形状有如锯齿一般，故称之为 Zigzag 扫描，其中第一个扫描的 DC 值位于左上角。

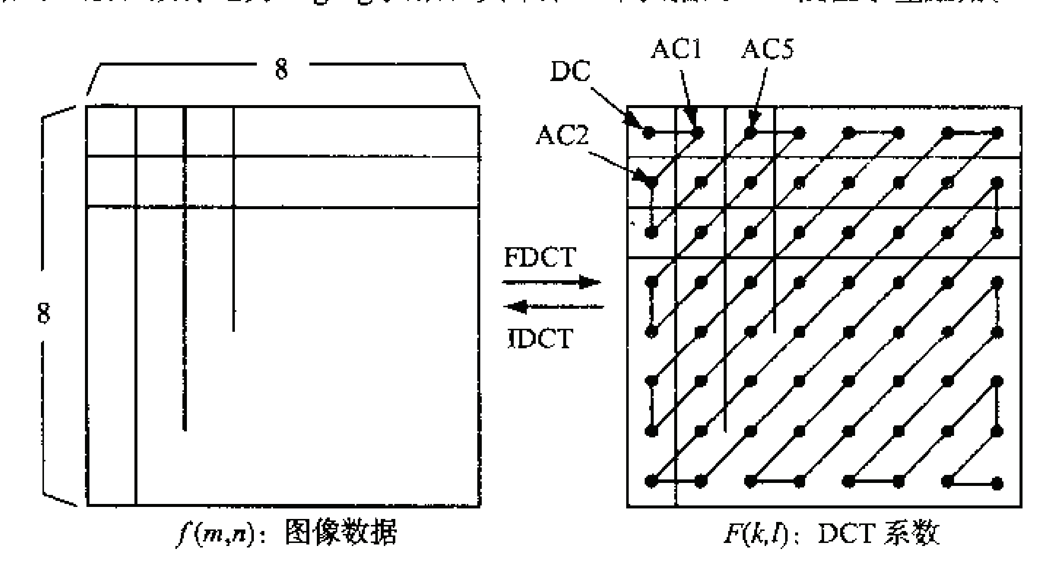


图 7.6-2　FDCT 与 IDCT 的关系

● **彩色图像压缩**

JPEG 可以适用的图像种类包括灰度与全彩色的图像。如 5.4-1 节所述，红、绿、蓝 (RGB) 是最简易的彩色模型，适合监视器三个电子枪的颜色显示。但对我们的感官视觉而言，这种模型未必是最自然的表现方式。这是因为和我们的眼睛比较有直接关系的是色彩的亮度及鲜

艳度等，所以 JPEG 舍 RGB 而采取 YCrCb 的色彩模型，其中 Y 代表亮度 (luminance)，而 Cr 与 Cb 则称为色度 (chrominance)。一般电视信号也是采用这种色彩模型，这可从电视上的控制键看出。RGB 与 YCrCb 这二种模型可依下面的方程式互相变换

$$
\begin{aligned}
Y &= 0.299R + 0.587G + 0.114B \\
Cr &= (R - Y)/1.402 \\
Cb &= (B - Y)/1.772
\end{aligned}
\tag{7.6-4}
$$

$$
\begin{aligned}
R &= Y + 1.402Cr \\
G &= Y - 0.344Cb - 0.714Cr \\
B &= Y + 1.772Cb
\end{aligned}
\tag{7.6-5}
$$

一般而言 R、G、B 都是以 8 个位来表示，亦即其数值范围在 0 到 255 之间，在变换成 Y、Cr、Cb 后，Y 的范围仍然是 0 到 255，但 Cr、Cb 则是 $-128 \sim 127$。因此在取 FDCT 之前，先将 Y 减去 128，使其范围也落在 -128 到 127 之间，一致的动态范围对后续处理较为方便。一个彩色图像有 Y、Cr、Cb 三个色彩成分，而灰度图像则只有亮度，也就是只有 Y 一个色彩成分。事实上 JPEG 并不限制图像有多少个色彩成分，但一般的应用上只有这两种色彩成分的组合。

由于 JPEG 是以一个 8×8 的块为单位来处理，因此对于每种色彩成分，都要切割成不相重叠的 8×8 块，再分别作 FDCT 的处理。由于人的眼睛对于 Y、Cr、Cb 三种成分的敏感度不同，例如眼睛对于亮度 Y 的感觉会比 Cr、Cb 来的敏锐，所以这三个成分在视觉上的重要性并不相同。由于 Cr 与 Cb 的重要性比较低，可以容忍的误差比较大，因此为了获得较大的压缩比，对于 Cr 与 Cb 的数据可以每两个点只取一点来处理，这种减少数据量的作法称为缩减取样 (subsampling)。如图 7.6-3 所示，每 16×8 的块就可以取其中一半成为一个 8×8 块；至于 Y 则不作缩减取样，否则会影响图像的品质。形成缩减取样的方式有两种：对于一个 16×16 的块而言，Y 取四个 DCT 块，Cr、Cb 则各取两个 DCT 块，这种格式称为 YUV422；如果 Cr、Cb 每四个点才取样一点则称为 YUV411 的格式，因为每 16×16 的块才取样成一个 DCT 块。

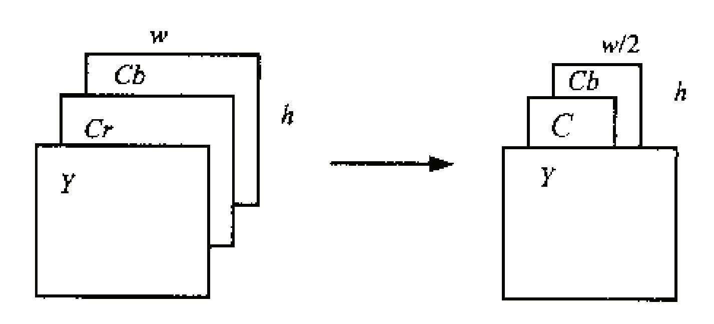


图 7.6-3　缩减取样

至于取样方法则有缩减取样及扩增取样 (upsampling)，如图 7.6-4 所示。此处的缩减取样是指要取样的该点和其左右 2 点以 2 : 1 : 1 的比重加起来。当数据要还原时，则在每两个取样点间加入一点，其值为两个取样点的中间值，这个过程就是扩增取样。缩减取样可减少很大的数据量，达到更高的压缩比。以 YUV422 而言，就使数据减少了 1/3，而 YUV411 的方式则使数据减少一半。

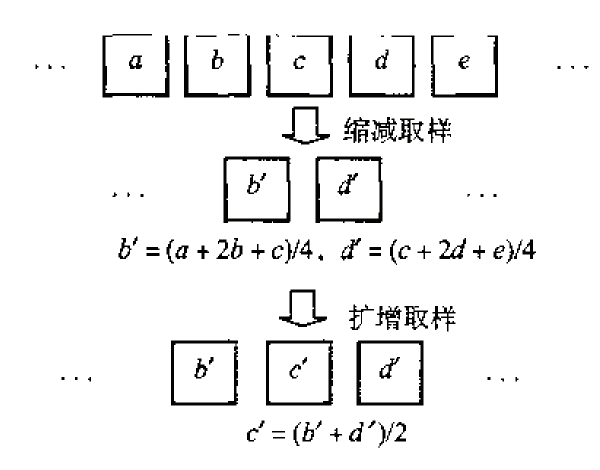


图 7.6-4　缩减取样与扩增取样的方法

● 量　化

量化是许多压缩技术中不可或缺的一环。由于许多经处理的数据是连续性的，例如在 FDCT 后所产生的系数值都是带有小数点的浮点数，必须要量化才真正可达到压缩的效果。JPEG 提供两个 DCT 系数的量化表，如图 7.6-5，每个表含 8×8 共 64 个值对应 64 个系数。量化的方式就是将每个系数除以表中相对应的值，取其最接近的整数

$$F_q(k,l) = \mathrm{round}\left(\frac{F(k,l)}{Q(k,l)}\right) \tag{7.6-6}$$

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

Luminance 的量化表

17	18	24	47	99	99	99	99
18	21	26	66	99	99	99	99
24	26	56	99	99	99	99	99
47	66	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99

Chrominance 的量化表

图 7.6-5　JPEG 所提供的量化表

当解压缩时，则将系数乘上量化表的值。可想而知这必定会产生误差，因此量化表的值会影响图像品质及压缩比。由于亮度与色度的重要性不同，因此各有不同的量化表。JPEG 提供的表是经过大量图像测试的实验结果，对于大部分的图像都有很好的效果。从表中可以发现，高频系数所对应的量化值较大，其目的就是要把高频的部分变成几乎都是 0 或接近 0，这对后面以 RLE 为基础的编码会有很大的帮助。

JPEG 可以在压缩比与图像品质间作一个取舍，其方法就是改变量化表的值。例如如果把量化表的值都放大一倍，将有更多的系数被量化成 0，而达到更高的压缩比，但品质一般而言也会降低。

● 编　码

当 DCT 的系数量化之后，就要将这 64 个系数编码。编码的方式可以用算术编码法

(arithmetic coding) 或使用霍夫曼编码法，虽然前者的效果会好一点，但一般都是采用较简单的后者。对每个 DCT 块而言，首先将 DC 的值减掉上一个 DCT 块的 DC 值，这个差值通常不大，因为相邻 DCT 块的颜色或灰度变化不大。以图 7.6-6 为例，DC 差值为 5。这个数字将被表成 (SS), VV，其中的 VV 表示该差值，即 VV = 5，而 SS 则表示要用多少位来表示该值。VV 是以可变长度的整数 (variable length integer, VLI) 来表示，由于 5 表示为二进位的 101，只要用 3 位来表示，因此 SS = 3。表 7.6-2 列出不同范围的整数所需的位数目，例如 – 3、– 2、2、3 四个数字都只要用 2 位就可以表示，分别表示为 00、01、10、11。

<table>
<tr><td rowspan="8">–45</td><td>–40</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>–2</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

图 7.6-6 DCT 系数的编码

表 7.6-2 VLI 的大小

大 小	范 围	
1	-1	1
2	-3,-2	2,3
3	-7...-4	4...7
4	-15...-8	8...15
5	-31...-16	16...31
6	-63...-32	32...63
7	-127...-64	64...127
8	-255...-128	128...255
9	-511... -256	256...511
10	-1023...-512	512...1023
11	-2047...-1024	1024...2047

对于 AC 的部分，则依照 Zigzag 顺序由 AC1 到 AC63，找到每一个非零的 AC 值，就将其表示成 (NN/SS),VV，其中 NN 表示该 AC 值前 0 的个数，而 SS 与 VV 则与先前的定义一样，但这里的 SS 不得为 0。例如图 7.6-6 中的 AC3 = – 2，在其前面的 AC1 和 AC2 均为零，故表示成 (2/2), – 2。当 NN 超过 15 时，则以一个 (15/0) 来表示有 16 个连续的零，例如图中的 AC21 = 1，原本应表示为 (17/1),1，现在则表示成 (15/0) (1/1),1。另外若有一串的零延伸到 AC63，不论零的个数有多少，一律都用一个 (0/0) 来代表块编码结束。根据上述法则，图 7.6-6 的系数可表示成：(3), 5 (2/2), – 2 (15/ 0) (1/1), 1 (0/0)。

这些符号最后均以霍夫曼编码得到实际的压缩数据。JPEG 提供数个霍夫曼编码表，如

表 7.6-3 中所列的为其中一部分，这些表是根据许多图像测试得来的平均结果，对大多数的图像而言都有不错的效果。编码表一般有四个，分别给亮度的 DC 与 AC，以及色度的 DC 与 AC 使用，这是因为量化后的亮度与色度系数的概率分布不同，为了得到较好的效果，因此使用不同的表。编码时，表示 DC 的 (SS) 应该对照 DC 编码表的 (SS) 而得到对应的码；同样的，表示 AC 的(NN/SS)则对照 AC 编码表的(NN/SS)而得到对应的码。由于 NN 与 SS 都不超过 15，因此可以各用 4 位组成一个索引值。至于 VV 的部分则是以 SS 位来表示，如果 VV 大于零则直接取其最低的 SS 位，若小于零则记录 VV－1 的最低 SS 位，若是 VV 等于零则不加以记录。

表 7.6-3 JPEG 所提供的几个霍夫曼编码表

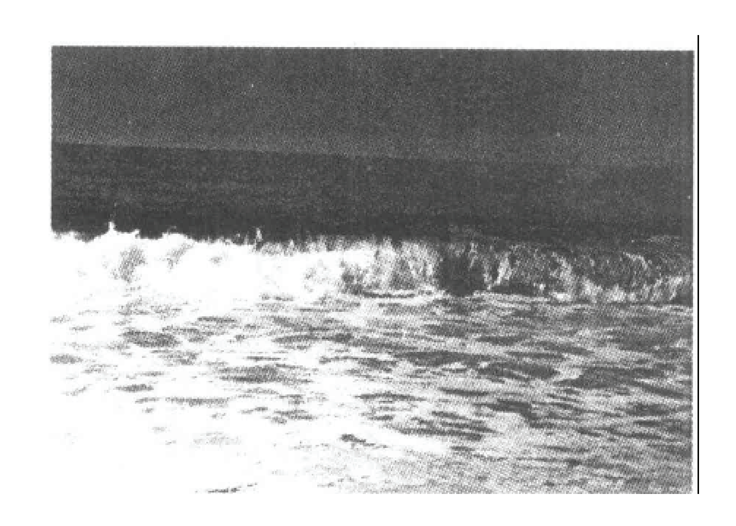
(3),5 (2/2),-2 (15/0) (1/1),1 (0/0)

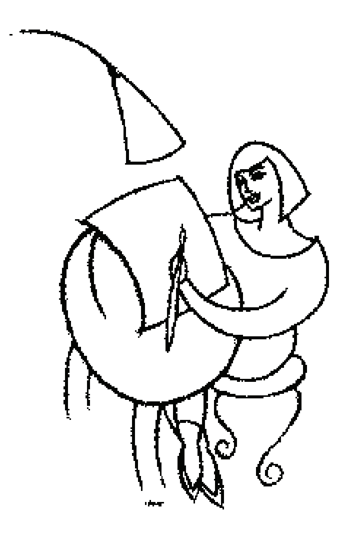
==>100 101 111110001 01 11111111011 11100 1 1010

亮度与色度的 DC 值			亮度与色度的 AC 值		
(SS)	亮度码	色度码	(NN/SS)	亮度码	色度码
0	00	00	0/0	1010	00
1	010	01	0/1	00	01
2	011	10	0/2	01	100
3	100	110	0/3	100	1010
4	101	1110	0/4	1011	1000
5	110	11110	0/5	11010	11001
6	1110	111110	0/6	1111000	111000
7	11110	1111110	0/7	11111000	1111000
8	111110	11111110	0/8	1111110100	111110100
9	1111110	111111110	0/9	1111111110000010	1111110110
10	11111110	1111111110	0/A	1111111110000011	111111110100
11	111111110	11111111110	1/1	1100	1011
			1/2	11011	111001
			1/3	1111001	11110110
			⋮	⋮	⋮
			F/9	1111111111111101	1111111111111101
			F/A	1111111111111110	1111111111111110

以图 7.6-6 的几个值为例，假设该 DCT 块是属于亮度的色彩成分，那么对照编码表就可以得到压缩后的数据。图中显示，原本一个 DCT 块有 64 个字节，压缩后只有 37 位，约为 4.6 个字节，故有约 14 倍的压缩比。事实上，有许多的 DCT 块在量化后几乎只剩下 DC，而 AC 全部为零，此时压缩比会更高。

最后我们比较一下 RLE 与 JPEG 的压缩效果，以及 JPEG 压缩法在不同压缩程度下图像还原后的差异程度。所采用的原图像显示于图 7.6-7 中。





(a) 自然图像　　(b) 手绘图



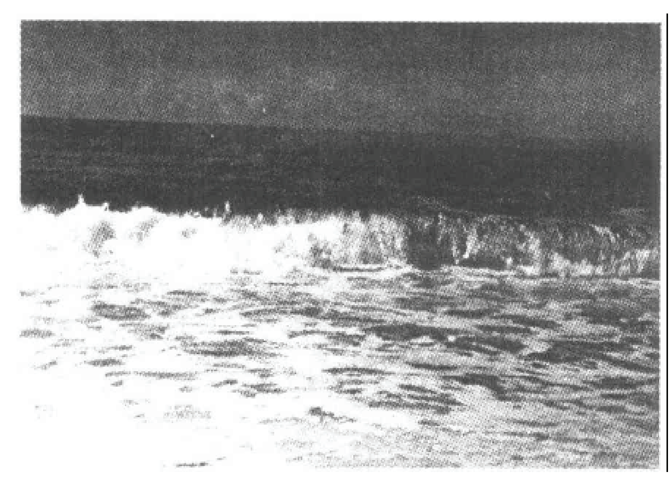
图 7.6-7　原始图像

这里所使用的压缩效率，是指被压缩掉的大小除以原本大小的百分比，因此百分比愈高代表压缩效能愈佳。由表 7.6-4 中显示，RLE 压缩法应用在手绘图这种较简单的图像上时，会有较佳的压缩效率，而 JPEG 则对两者都有非常好的压缩效果。

表 7.6-4　比较 RLE 及 JPEG 的压缩效率

	Sea	Painting
原图	344,785	128,592
RLE	318,720(7.5%)	23,380(81%)
JPEG(10)	39,819(88%)	8,695(91%)
JPEG(20)	27,645(92%)	6,751(94%)
	单位：字节	

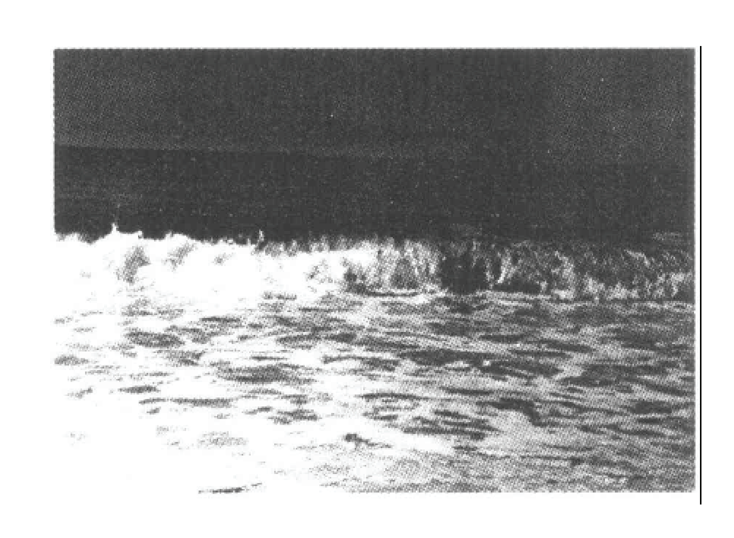
图 7.6-8 及图 7.6-9 显示 JPEG 分别采取压缩参数 10 及 20 后，图像还原的结果。在手绘图图像中，由于有较清楚的边缘存在，所以在经过 JPEG 压缩后，在边缘处会有模糊的现象，而这一缺点在另一张自然图像上较不易察觉。





(a) 自然图像　　(b) 手绘图

图 7.6-8　指定压缩参数 10 的 JPEG 压缩还原结果



(a) 自然图像



(b) 手绘图

图 7.6-9　指定压缩参数 20 的 JPEG 压缩还原结果

7.7　动态图像压缩

7.7-1　视频图像压缩标准 MPEG-2

由于通讯科技的发展及超大规模集成电路技术的日新月异，使人类在生产消费与休闲娱乐上有革命性的转变。为了使每一终端用户能享受通讯服务，单以原有的通讯媒介是无法承受的。除了布置宽带光缆以取代现有的铜线回路外，降低数据量是解决有限频道资源的有效办法。而各类信息中，又以视频图像所需要的数据量最大，因此视频图像压缩为现今媒体通讯的关键技术之一。从 CCITT 的 H.261 到 MPEG 压缩，无论是在画质、解析度、动作连续性及错误的遮隐能力都有很大的进步，但是基本上它并未满足在广播或 HDTV 或有线电视等方面的应用需求。这些系统不仅要求更高的画质及解析度，同时还要能通过各种网络做不同的服务，因此才有 MPEG-2 标准的制定。本节将探讨国际标准化的视频图像压缩方法，特别是 MPEG-2 的视频图像编解码标准。

● **MPEG-2 特性**

MPEG-2 与 MPEG-1 的不同之处在于前者的语法结构至少提供了下列各种特征：

1. 提供逐行与隔行的扫描格式

一般电视的信号属于隔行 (interlaced) 扫描式，而有些计算机及视听器材属于逐行 (progressive) 扫描式。MPEG-2 能针对隔行式或逐行式动态图像的特性获得最佳的编码效果。

2. 提供 3：2 变换功能

这使得电影 (每秒 24 个画面) 到电视信号 (NTSC 每秒 30 个画面) 之间的变换有了直接面简便的途径。

3. 提供移动型的视窗显示

用一个低解析度的电视如何收视高解析度画面的节目？可以用把视窗的位置和大小编入位码流来解决。

4. 画面品质的可调度更大

因为位率的弹性加大了，节目的制作者可充分运用可利用的位来调整画面的品质。同时因量化矩阵和量化阶数的选择变化增多，因量化而产生的误差相对可以做得比较小，使品质提高。

5. 随机撷取

一般所指的随机撷取，着重于储存介质上数据的随机撷取。将此观念用至视频编码上，即每一幅画面都能在规定时间内被解码或撷取，在作法上则是将编好码的位流切割成互相独立的区段，每一区段都由一I画面领头，因此可以独立解码。这样的设计使得视频编码在广播电视上的应用更为便利，因此基本上使用者每变换一次频道都等于是另一随机撷取的开始。

6. 高低复杂度的解码器

由于MPEG-2是向下兼容，因此一个MPEG-2解码器完全可解任何MPEG-1位流，但是MPEG-1解码器无法解任何MPEG-2位流。在MPEG-2 profile和层次(level)的设计之下，MPEG-1位流可以被包含在低层次之下被MPEG-2解码器所解码。

7. 差错的掩盖

画面组的设计虽然可以把错误的传播限制在其所属的画面组内，但若受损的画面是I画面，则收视者极有可能会感受到多至半秒的不良画面，这是高价值高品牌的产品所不允许的。MPEG-2利用差错掩盖位移向量来修补损坏的宏块，使差错的扩散减至使收视者不易察觉的程度。

7.7-2 MPEG-2标准的原理

● MPEG-2标准

MPEG委员会于1988年成立，并与Expert Group for ATM Video Coding于1990起合作制定出ISO/IEC 13818国际标准，也就是MPEG-2标准。其中包含：

1. ISO/IEC 13818-1 系统

此部分定义一个整合图像与声音的多工结构及图像与声音同步所需的时序信号的表示方法。在系统中定义了两种数据流：传输数据流(transport stream)及节目数据流(program stream)，其中传输数据流主要应用在网络有线电视，节目数据流则用于数字储存媒体，如CD-ROM。

2. ISO / IEC 13818-2 图像部分

主要是定义经过压缩的图像数据的形式及解压缩所需的过程。

3. ISO / IEC 13818-3 音效部分

定义音效数据压缩的编码方式。

4. ISO / IEC 13818- 4 兼容性

定义出一系列的程序以测试某一数据流是否与标准相符。

● **MPEG-2 数据结构**

MPEG-2 图像数据结构共六层 (见图 7.7-1)，除了宏块及块外其余各层都有独特的起始码加以辨认。视频序列是最上层，内含数据包括画面大小、量化矩阵及每秒画面数目等，基本上每一层均由标头 (header) 及有关下一层的信息所组成。

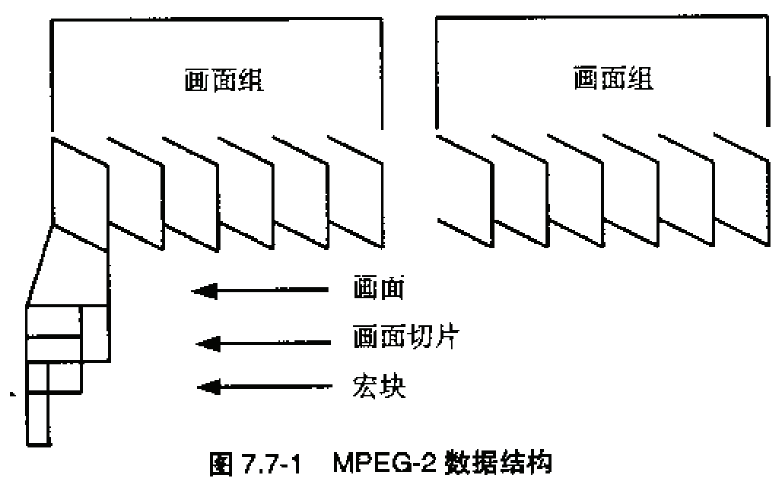


图 7.7-1　MPEG-2 数据结构

在视频序列之下就是所谓画面组，它使我们可以针对某一视频序列内的所有画面做随机存取，每一个画面组一定是以 I 画面 (I picture) 为开始。画面层即是视频序列内每一张个别的画面。在 MPEG-2 内定义三种画面：

I 画面 (内部编码，intracoded)

P 画面 (预测编码，predicted)

B 画面 (双向预测编码，bi-directionary predicted)

其中 I 画面是参考本身的数据来做编码；P 画面是根据过去的 I 画面来编码；B 画面则是参考前面及后面的 I 画面或 P 画面来编码。由于 B 画面需要靠双向的 I 画面及 P 画面来编码，所以会造成如图 7.7-2 般编码与播放顺序不同的情形。

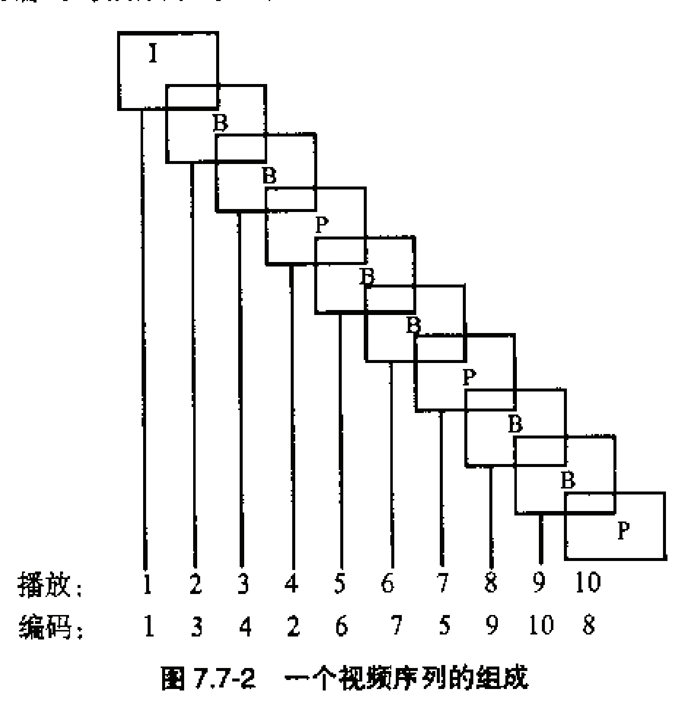


图 7.7-2　一个视频序列的组成

每一画面又可分成若干个切片，每一个切片包含若干个宏块，由图 7.7-3 可看出之间的关系。

MB	MB	MB	…	MB	MB	MB
MB	MB	MB	Slice	MB	MB	MB
MB	MB	MB	…	MB	MB	MB
MB	MB	MB	Slice	MB	MB	MB

图 7.7-3　画面切片 (Slice) 与宏块 (MB) 的关联

每一个宏块包含数个8×8像素区块。每一个像素由亮度 (Y) 及色度 (Cb, Cr) 组成，根据不同的 Y、Cb、Cr 比例，图 7.7-4 说明有所谓的 4∶2∶0、4∶2∶2 及 4∶4∶4 三组。4∶2∶0 是指针对四个像素，利用四个 Y 及两个 Cb 及 Cr (各一个) 数据来表示，即四个 Y 共用一个 Cb 及 Cr；4∶2∶2 即四个 Y 共用两个 Cb 及 Cr；4∶4∶4 则表示每一个 Y 有一个 Cb、Cr。其中数据越多 (如 4∶4∶4) 越适合专业的图像工作室，因为它们通常有较高品质的需求。

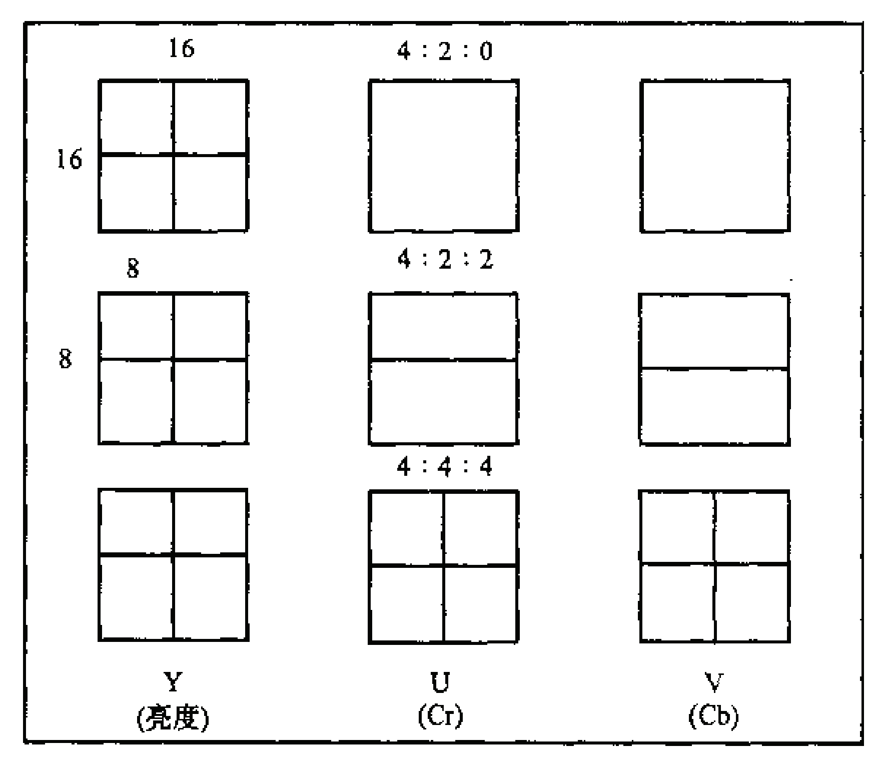


图 7.7-4　不同 YUV 比例

● **MPEG-2 图像解码器**

图 7.7-5 是一个典型的 MPEG-2 图像解码器的应用环境。从图 7.7-5 我们可看出一个

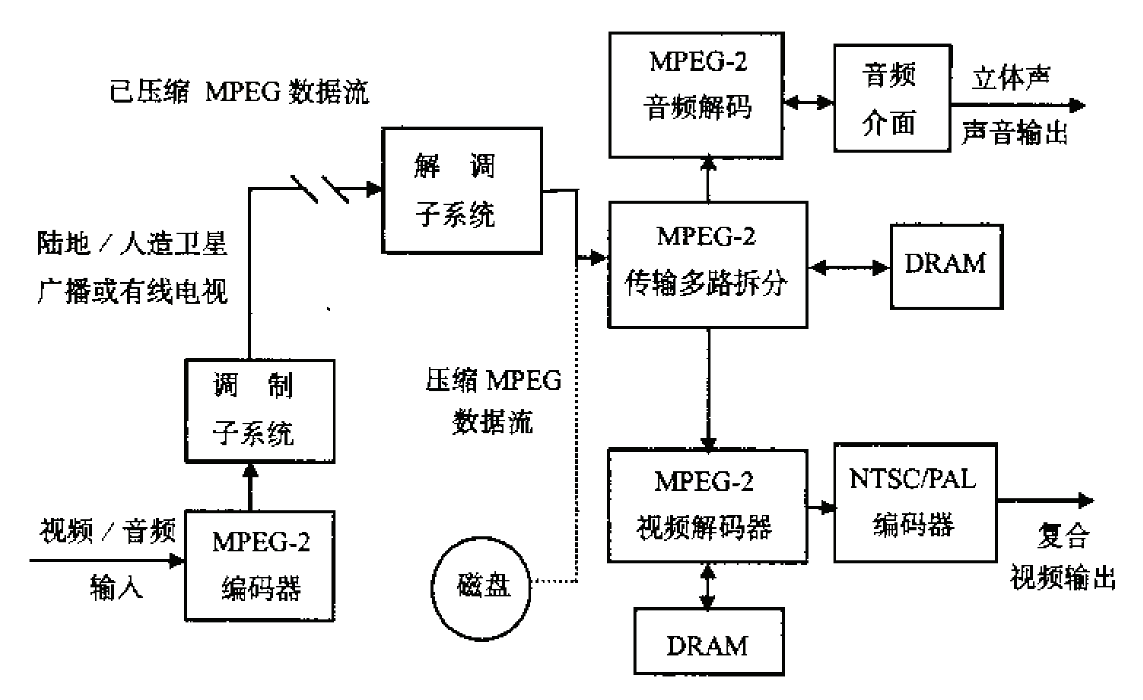


图 7.7-5 MPEG-2 的应用环境

MPEG-2 图像解码器至少应：

(1) 与主处理器作沟通，接受经过分离的图像压缩数据。

(2) 解压缩。

(3) 与存储器连接，存取压缩／解压缩过的图像数据。

(4) 与视频编码连接，提供解压缩过的数字数据给编码器以产生模拟的视频信号，输出至电视或监视器。

图像解码器的方块图，通常包含了下列几个单元：① 内部微处理器；② 解码器； ③ 反量化器；④ 反离散余弦变换；⑤ 动态补偿；⑥ 存储器介面；⑦ 主处理器介面；⑧ 显示控制器介面。

内部微处理器为图像解码器的控制中心，它从外部的主处理器接受命令并提供目前解码器的状态，并负责解码器内部其他单元之间的同步工作及解释隐含在 MPEG-2 图像数据流标头的信息，以产生必需的参数及控制信号给其他单元。

MPEG-2 使用霍夫曼的编码方式，并在标准中定义了两组对照表 (MPEG-1 中只有一组)。霍夫曼解码器根据对照表，将压缩的数据变换成一串游程长度编码。

此反量化器单元的主要功能包括：

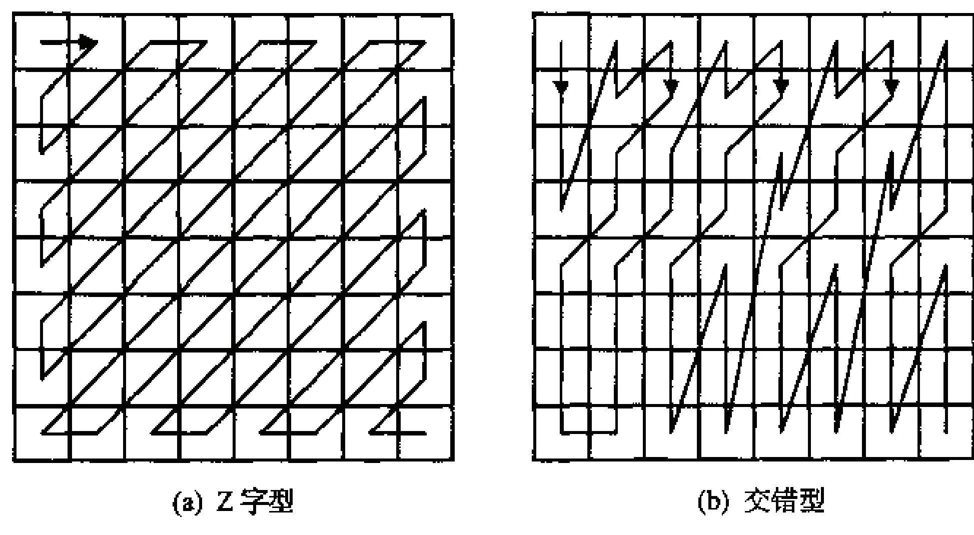
① 将游程长度编码转成真正的码。如 30 21 22 11 33 → 000 11 22 1 333。

② 将变换出的数据用两种不同的扫描方式组成一个 8×8 的矩阵。如图 7.7-6 所示，其中(a)是 Z 字型扫描、(b)是交错型 (alternative) 扫描，而在 MPEG-1 中只定义了一种扫描方式。

③ 执行 DPCM 以求得此一 8×8 的 DC 系数 (位于 8×8 矩阵左上角的系数)。

④ 将此一 8×8 矩阵(左上角 DC 除外)乘上一个量化因子以得到一个还原的反离散余弦变换系数矩阵。

⑤ 反离散余弦变换：将还原的系数矩阵做处理。



(a) Z 字型　　　　(b) 交错型

图 7.7-6　扫描方式

在前面提过有所谓的 P-画面及 B-画面，基本上，这类画面并不是真正根据自己画面的内容加以压缩，而是在过去或未来的画面中去找寻一个与它相符的宏块，然后算出运动向量(从目前画面中运动到过去 (未来) 相符的宏块所需的向量)，如图 7.7-7 所示。运动补偿即是根据运动向量来计算出真正的宏块。

其余如存储器介面、主处理器介面、显示控制器介面则视各个厂商设计有所不同，但是大同小异，主存储器介面目前大部分使用 IBM 的 Power PC、Intel 386EX、Motorola 68k、……显示控制器介面则有 YUV 输出及 GRB 输出两种介面。

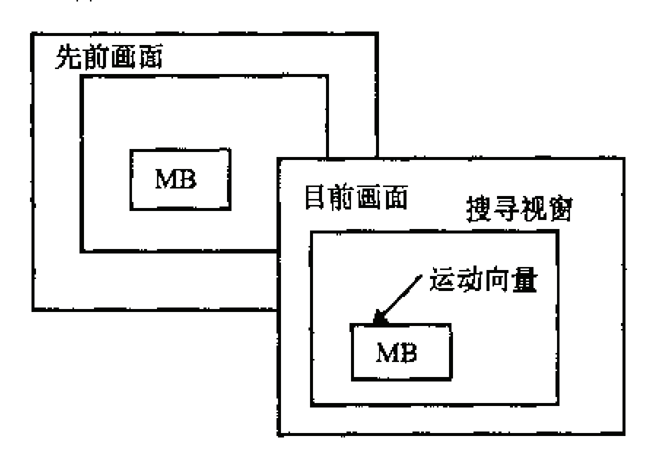


图 7.7-7　区块对比运动补偿

7.8　以小波变换压缩的实例

7.8-1　导　论

本节所使用的压缩方法是将图像小波变换所得的小波系数做向量量化，同时使用金字塔

型多解析度结构。以小波变换为基础的多解析度树状分解如图 7.8-1 所示。其中 L 与 H 分别代表沿图像的行或列取低与高频成分的运算。

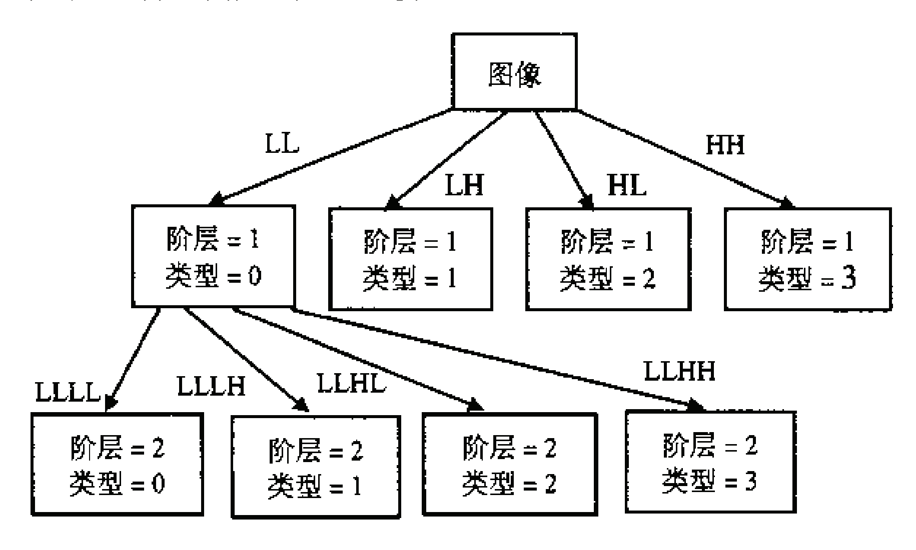


图 7.8-1　将图像作二层分解示意图

将小波变换应用到图像上并未降低欲压缩数据的数据量。真正的压缩是发生在将系数量化时。量化可大致分成向量量化 (vector quantization, VQ) 与纯量量化 (scalar quantization, SQ) 两种，在本书第三章中曾提到，此处讨论重点则在于如何与小波系数结合。

7.8-2　量　化

● **向量量化准则**

小波变换与向量量化的结合，如图 7.8-2 所示，其中的多解析码本是指对不同阶层或类型的小波系数分别给向量维度或码本大小可能不同的向量量化。最常用的码本设计方法是 3.3 节所介绍的 LBG 算法，其中初始码本 $\hat{A}_0$ 的选择可用以下的方式。

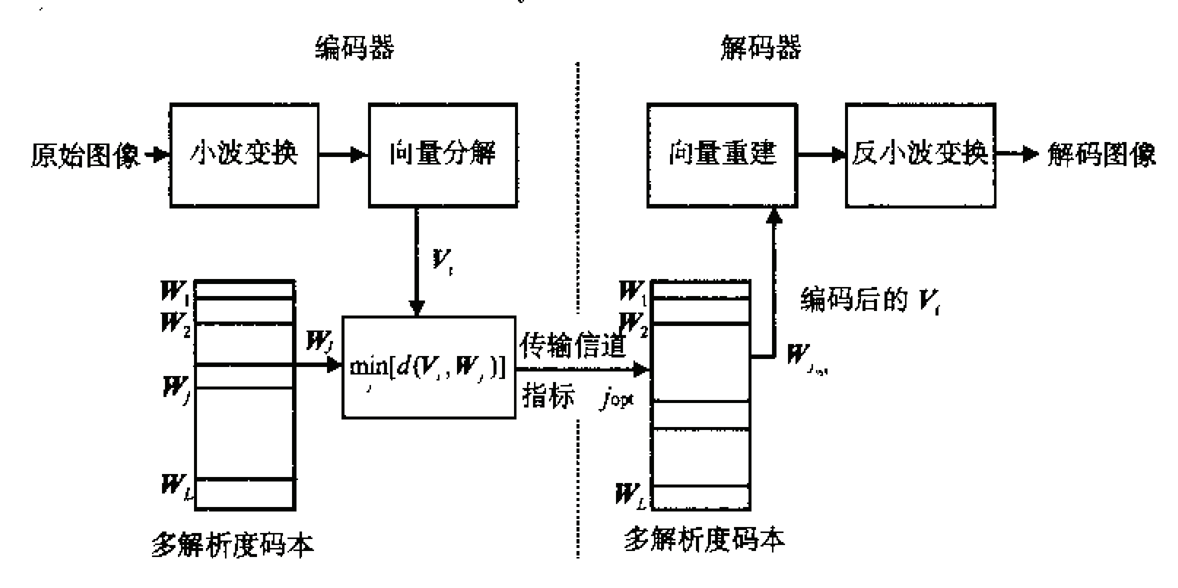


图 7.8-2　编码 / 解码流程方块图

(0) 起始状态：设 $M=1$，定义 $\hat{A}$ 为全部输入向量的中心点。

(1) 令含 M 个向量 $\{\boldsymbol{y}_i; i=1,2,\cdots,M\}$ 的码向量为 $\hat{A}_0(M)$，接着将每个向量 $\boldsymbol{y}_i$ 分割成两个近似的向量 $\boldsymbol{y}_i+\varepsilon$ 和 $\boldsymbol{y}_i-\varepsilon$，此处 ε 为一固定扰动向量。$\{\boldsymbol{y}_i+\varepsilon, \boldsymbol{y}_i-\varepsilon, i=1,2,\cdots,M\}$ 的集合 $\tilde{A}$ 有 $2M$ 个向量。用 $2M$ 取代 M。

(2) 令 $\hat{A}(M)=\tilde{A}(M)$。M 与 N 是否相等？如果是，则停止，此时 $\hat{A}$ 即为所求。如果不是，在 $\hat{A}(M)$ 上执行 M 阶量化器的 LBG 算法并且返回子步骤 (1)。

- **与纯量量化的比较**

比较纯量量化和向量量化时应考虑二方面：压缩比的达成及图像品质，表 7.8-1 比较了 SQ 和 VQ 的差异。在 SQ 方面，假设要量化的取样点有 Gaussian、Laplacian 和 Gamma 函数概率密度分布，从对应的 Lloyd-Max 量化器中选出结果最好的列于表中。一般而言，Gamma 函数最接近小波的概率密度分布。

表 7.8-1　应用在 Lena 图像系数上的类型 1，阶层 3 和 4 的纯量与向量量化的结果

bpp	阶层	类型	N	k	L-M SNR	LBG SNR
1.0	3	1	2	1	1.68	2.32
1.0	3	1	4	2	-	3.75
1.0	3	1	16	4	-	5.47
1.0	4	1	2	1	2.10	2.45
1.0	4	1	4	2	-	3.88
1.0	4	1	16	4	-	6.38
2.0	3	1	4	1	6.40	6.82
2.0	3	1	16	2	-	8.67
2.0	4	1	4	1	6.83	7.33
2.0	4	1	16	2	-	9.93
3.0	3	1	8	1	7.89	11.85
3.0	3	1	64	2	-	13.75
3.0	4	1	8	1	9.09	12.48
3.0	4	1	64	2	-	16.80

注：表中 N 为码本大小，k 为每一向量的维度。在 LBG 一例中，位率 Rbpp 由 $N=2^{kR}$ 计算而得。L-M SNR 代表以 Lloyd-Max 量化器所得的信噪比；LBG SNR 则代表以 LBG 训练法所得的向量量化器所得的信噪比。

此外，$k=1$ 时，VQ 变成 SQ，但由表中发现此时 VQ 的表现仍较 SQ 为佳。其中的原因很可能是实际的小波系数密度分布与 Gamma 函数有些许差异，而 LBG 则仍适应真实的小波系数。

- **小波系数的统计分布**

每一类型小波系数的概率密度函数对于了解在不同解析度时的行为非常重要。这个结果可用 Lena 的小波系数来论证，其他图像也有类似结果。图 7.8-3 将阶层 2 类型 1 的小波系数的概率密度函数以 Gaussian、Laplacian 和 Gamma 函数来近似

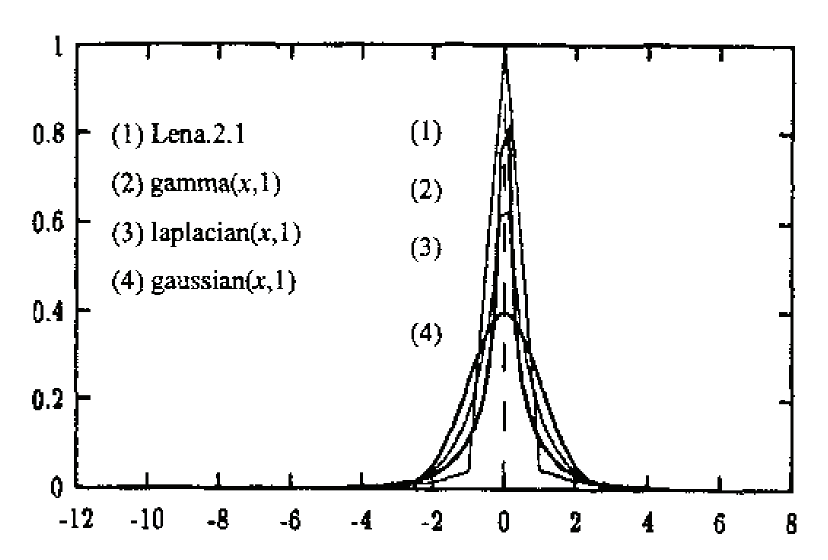


图 7.8-3　用 Gaussian、Laplacian 和 Gamma 函数表示阶层为 2 且类型为 1 的小波系数的概率密度函数

$$\text{Gaussian}\left(x,\sigma_x\right)=\frac{1}{\sqrt{2\pi\sigma_x^2}}\exp\left(-x^2/2\sigma_x^2\right) \tag{7.8-1}$$

$$\text{Laplacian}\left(x,\sigma_x\right)=\frac{1}{\sqrt{2\sigma_x}}\exp\left(-\sqrt{2}|x|/\sigma_x\right) \tag{7.8-2}$$

$$\text{Gamma}\left(x,\sigma_x\right)=\frac{\sqrt[4]{3}}{\sqrt{8\pi\sigma_x|x|}}\exp\left(-\sqrt{3}|x|/2\sigma_x\right) \tag{7.8-3}$$

检验灰度图像的小波系数可发现，除低解析度外，在同一阶层或不同阶层的所有类型均有类似的分布，如图 7.8-4 所示。

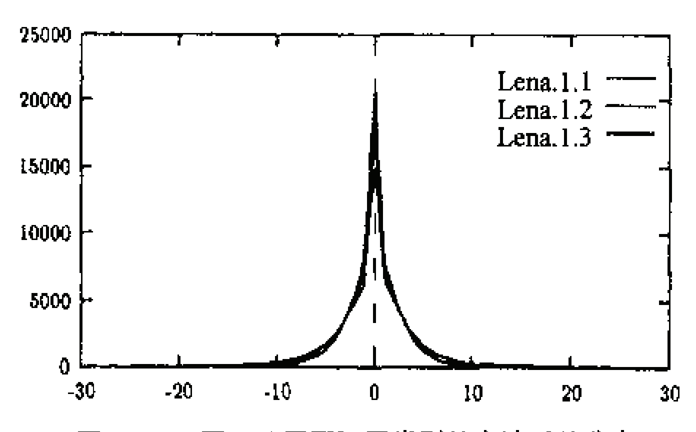


图 7.8-4　同一阶层下不同类型的小波系数分布

表 7.8-2 显示 Lena 图像的小波系数的特性，包括各阶层与类型的最大 (Max) 与最小值 (Min)，平均值 (Avr) 与方差 (Var)，以及对平均值与方差归一化的结果。表中显示方差随阶层数的增加而增加，表示愈低频的部分愈重要。此外，从表 7.8-2 最后二栏可看出经归一化后，不同阶层及类型均有类似的结果，这表示可以用特性相近的量化器对其量化。最后，阶层 4 类型 0 的意思是指最低频或最平滑的部分。

表 7.8-2　Lena 图像的小波系数的统计特性

阶层	类型	Min	Max	Avr	Var	(Min-Avr)/Var	(Max-Avr)/Var
1	1	-100.8	106.3	0.13	8.090	-12.4	13.1
1	2	-53.28	81.37	0.03	5.057	-10.5	16.0
1	3	-34.07	41.33	0.00	3.136	-10.3	12.4
2	1	-343.3	207.8	0.23	26.65	-12.8	7.78
2	2	-198.4	157.6	0.16	15.65	-12.6	10.0
2	3	-113.9	137.7	-0.14	12.17	-9.34	11.3
3	1	-479.6	587.8	-2.16	68.05	-7.01	8.66
3	2	-324.2	257.8	-0.34	38.74	-8.36	6.66
3	3	-256.5	315.1	0.13	35.85	-7.15	8.78
4	1	-931.6	1070	-4.46	196.1	-4.72	5.48
4	2	-593.8	682.2	-3.61	98.13	-6.01	6.98
4	3	-484.5	721.0	0.39	89.85	-5.39	8.02
4	0	-1743	1609	-155	780.5	-2.03	2.26

● **能量分布**

从表 7.8-2 及图 7.8-4 可以看出大部分系数都非常小且分布在一个以原点为中心的窄小动态范围内，且较高阶层包含有较小系数及方差。在表 7.8-2 及表 7.8-3 中，我们可以看出每一阶层中，类型 1 和 2 均比类型 3 含有较多能量和较大系数及方差。

表 7.8-3　Lena 图像之的小波系数的能量分布

阶　层	类　型	
1	1	0.563
1	2	0.220
1	3	0.094
阶层 1 的总和		0.877
2	1	1.528
2	2	0.527
2	3	0.318
阶层 2 的总和		2.373
3	1	2.492
3	2	0.807
3	3	0.691
阶层 3 的总和		3.99
4	1	5.179
4	2	1.298
4	3	1.088
阶层 4 的总和		7.565
4	0	85.19

此外，阶层 1 比阶层 2 的能量低且系数也较小，阶层 2 又比阶层 3 的能量低及系数小，依此类推。

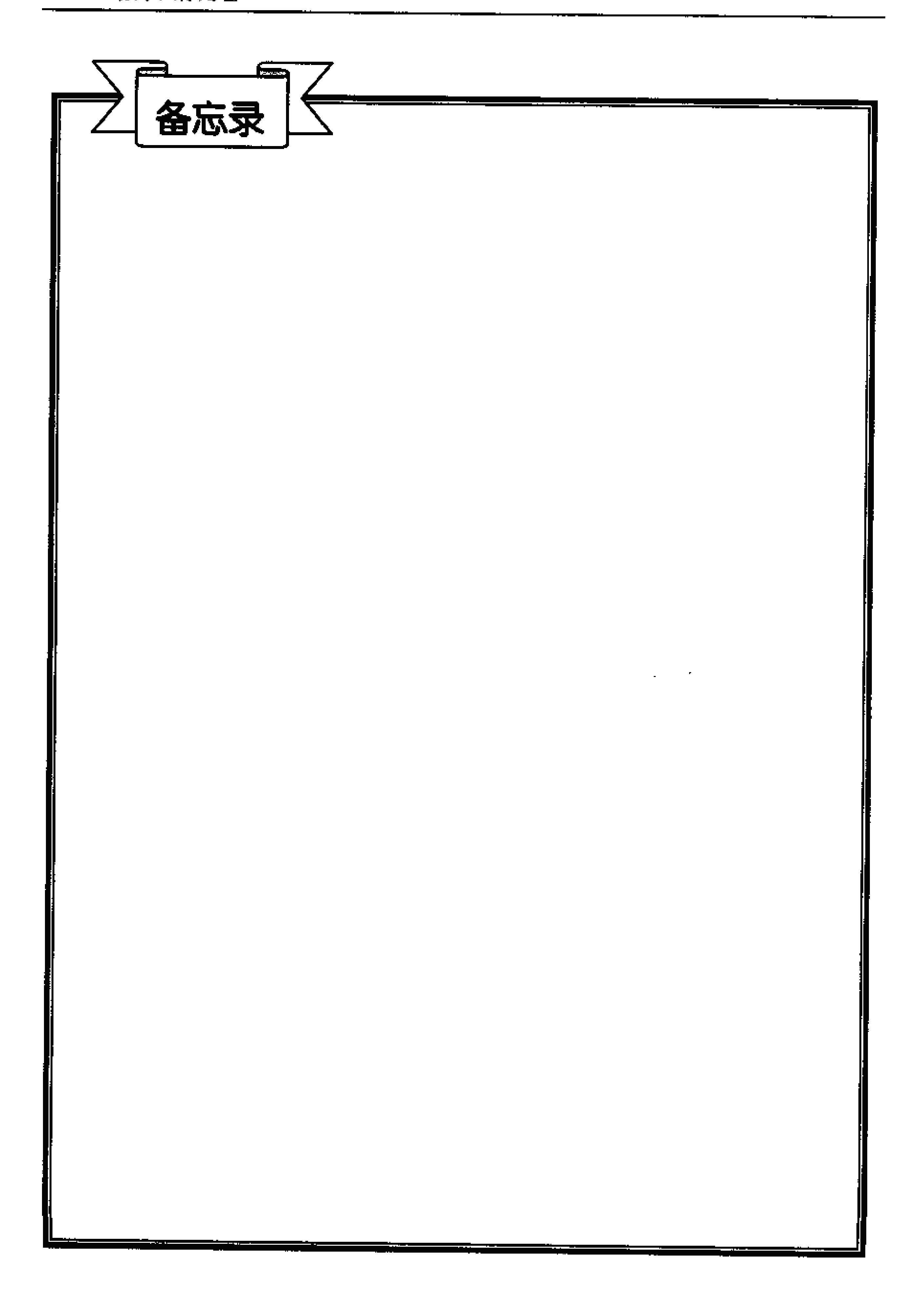
本节所有数据及内容均参考 Arerbuch et al. [1996]，对本课题有兴趣而欲知细节者可自行阅读该论文。

习　题

1. 设计一个 DCT 变换的编解码器如下：
   (1) 图像分割成不重叠的 8×8 方块，分别进行 DCT。
   (2) 按 Zigzag 顺序只保持 DC 及 AC1 ~ AC5 共 6 个系数，其余均取为零。
   (3) 对各系数进行直方图的统计，估测出六个概率密度函数 (设为高斯) 的参数 (即平均值与偏差量)。
   (4) 以第三章的方法为各系数设计最佳的纯量量化器 (位数自选)，并对 DCT 系数进行量化。
   (5) 进行反量化及 IDCT，最后重整出整幅图像。
   (6) 计算压缩比及重建图像的 PSNR。
2. 考虑一个 8 位图像源之熵的估测。设此信号源产生如下的像素值：

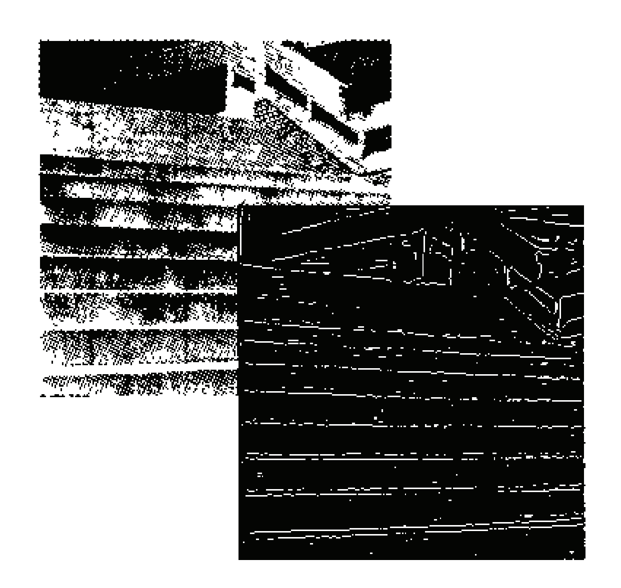
20	20	20	45	60	60	172	172	172
20	20	20	45	60	60	172	172	172
20	20	20	45	60	60	172	172	172
20	20	20	45	60	60	172	172	172

   (1) 假设各像素值为统计独立，并视一个像素值为一个字元符号，求所估测之熵。
   (2) 将 (20,20), (20,45), (45,60), (60,60), (60,172), (172,172) 视为 6 个不同的字元符号，重新求此字元符号的熵。
   (3) 取相邻 (左、右) 两像素值的差 (右减左) 为字元符号，求其熵。
3. 将上题的三个子题各以霍夫曼编码，计算该码的平均长度及编码效率。
4. 将一个小图像 (16×1) 以 PCX 及 CUT 的 RLE 进行编解码，比较其压缩比。
5. 模拟 JPEG 的变换与量化的程序(不需要编 0 与 1 的码)。
   (1) 用图 7.6-5 的量化表及将其内容一律乘上 2 倍的量化表。
   (2) 统计上述两种状况下量化后系数为零的个数，个数愈多通常表示编码后的压缩比愈高。
   (3) 依各自的量化表进行反量化及反变换，求其 PSNR。
   (4) 由 (2) 及 (3) 你觉得压缩比与品质之间可能有何关系?
6. MPEG 与 JPEG 在编码上有何异同?
7. 设计一个小波变换的编解码器如下：
   (1) 进行一个如图 7.8-1 所示的二层小波变换得 7 个频带。
   (2) 对各个频带系数进行维度与码本大小均可有所不同的多解析度码本 VQ 编码。
   (3) 计算压缩比及 PSNR。


备忘录

第八章

图像分割



本章要讨论的图像分割 (image segmentation) 属于图像分析的中层处理，先前各章所介绍的一些图像技术则可归类为底层处理的方法，如图 8.0-1 所示。下一章的图像表示与描述仍属中层处理，最后一章的图像模式识别则属高层处理。

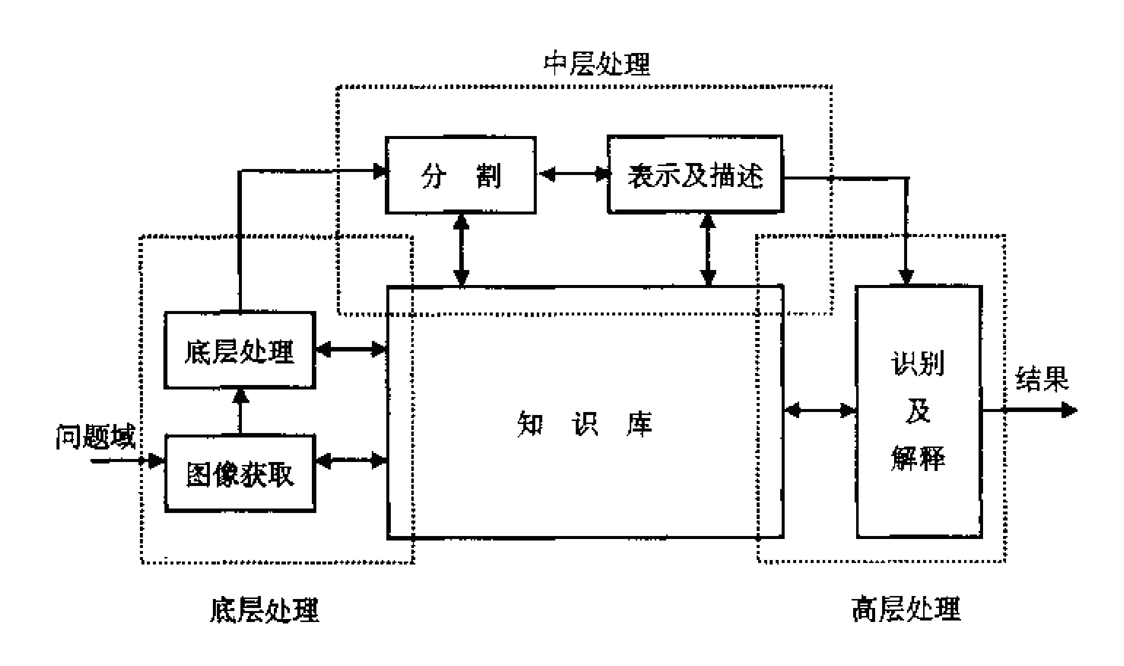


图 8.0-1 图像分析的结构

中层处理的任务就是提取和描述底层处理过后的图像，以便对于此图像中的物体更加了解其大小、位置、灰度级等，还可修饰物体的边界，对物体有更深的描述，甚至于将其与背景分离。我们必须了解一点：这些技巧及底层处理中一些去除干扰及去除模糊等技巧，都可视为做高层处理的准备工作。

8.1 导 论

我们现在先举一例子来说明图像模式识别 (pattern recognition) 的用途。假设现有一个将水果分类的系统，如图 8.1-1 所示。水果的种类包括樱桃、柠檬、苹果以及葡萄柚，这些水果经由输送带通过监视器，再将这些水果的图像送至处理机去辨认；而在处理器中根据这些水果的一些特征 (必须是能使电脑分辨出水果种类的特征) 来将水果分类，在此例中用的是大小及红色的程度，如图 8.1-2 所示。

根据图 8.1-2 我们可以很成功地将水果分开，这种系统将可以取代传统的人力，并且较为便利和快速。在图 8.1-2 中，我们以大小及红色的程度为辨认依据，那么这些信息要如何获取呢？第一步就是必须先将物体分离出来，并分析其大小及色泽；这里所用的方法正是我们接下来所要介绍的方法——图像分割法。

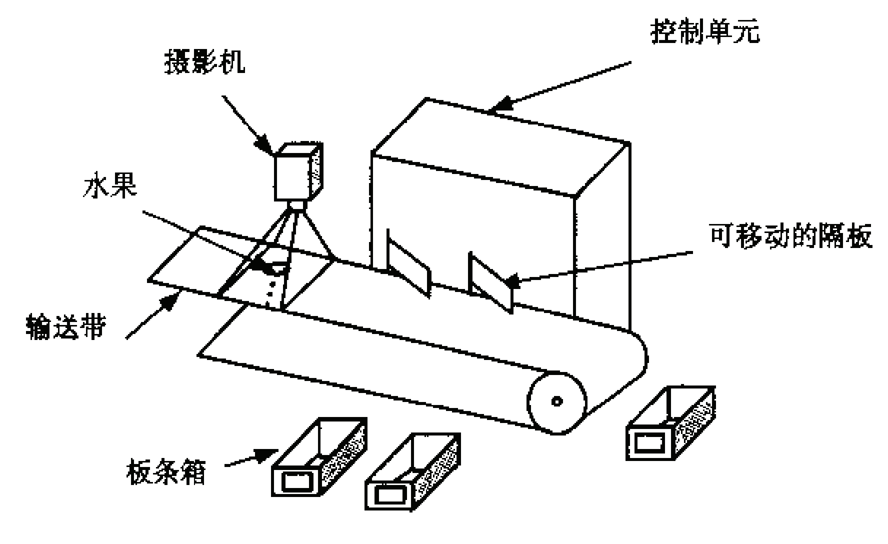


图 8.1-1 水果分类系统

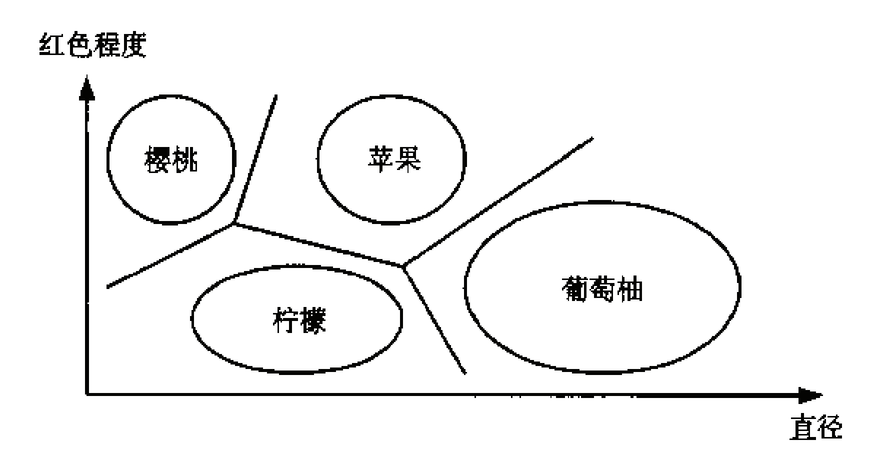


图 8.1-2 特征空间

8.2 图像分割处理

图像分割处理可以定义为将一数字图像分割成若干区域，而这些由像素组成的区域必须为各个相类似的像素相连而成。而所谓“相连”是指在连通 (connected) 的集合中，任两个像素之间有一条相连的路径，且此路径都是由集合中的像素所组成。至于相连的路径，则由像素间的连通性来决定。常用的连通性有下列两种：

① 四连通 (four-connectivity)：即对一像素而言，其与上、下、左、右的像素相连。

② 八连通 (eight-connectivity)：即除了上、下、左、右的像素外，还加上对角连线上的像素，故有八个像素与其相连。

图像分割大致有四种方法可以实现：

① 阈值法：这是一种以阈值为界区分物体与背景的简单方法。

② 区域法：这是一种仅认定某一个像素集合为一个物体或区域的方法。

③ 边界法：这是找寻存在两个区域之间边界的方法。

④ 边缘法：辨认像素的边界，并且将它们连结在一起找出物体边界的方法。

8.2-1 阈值法实现图像分割

阈值法主要靠设定阈值来分辨物体与背景。简言之就是在图像的灰度值中，选一适合的阈值；若小于阈值，则判为背景；大于阈值，则判为物体，因此这种方法很适合用于图像中的物体有一明显的区域分界且边界为封闭的情况。亦即，若所感兴趣的物体和背景之间的灰度值有明显差异时，则阈值将可完全地分割物体与背景。

- **整体阈值** (Global Thresholding)

这个方法是对整幅图像选定一固定灰度值，并以此值去做此图像的分类，以找出此图像的物体；此方法很简单，但因为步骤仅将灰度值分为高、低两类，因此只在物体与背景单纯且亮暗分明下，才会有好效果。

对此固定阈值的选择，一般采用的是直方图法。先将每一像素的灰度值按出现频率做一份数据统计并画成图 8.2-1 所示的图形。由图 8.2-1 看出这是一幅物体与背景分明的图像。我们可直接在交会点 (即边界) 之处选择其灰度值作为门限，由直觉上可知这就是最佳临界点 T_{opt} 的选择。

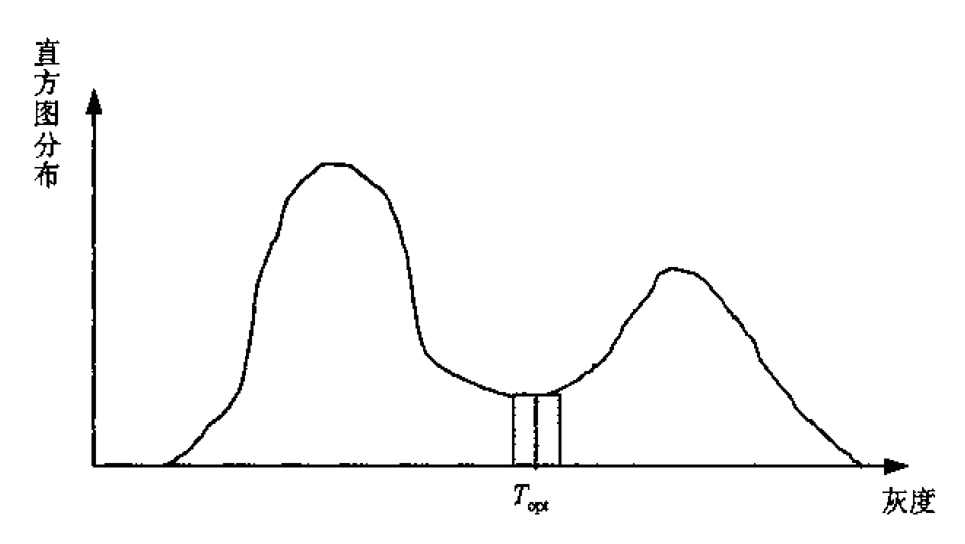


图 8.2-1 整体阈值的最佳选择

- **适应性阈值** (Adaptive Thresholding)

在很多的情况下，图像并不像上面所述那样单纯，背景上可能有很多的变化；在这种情况下，可能某些区域中所用的阈值，就不能用在别的区域上，当然最好的方法就是在不同的区域，有不同的阈值，亦即自适性的阈值。

为了适应图像局部的变化，阈值可依下面的步骤来求得：

步骤一：先将欲做分析的图像分成若干子图像，并就每个子图像做直方图的分析，去掉单峰 (unimodal) 的直方图，只记录具双峰 (bimodal) 的直方图。

步骤二：在背景的灰度值与物体的灰度值中，选择中间值来作为阈值。

8.2-2　区域法实现图像分割

设 R 代表整个图像区域，图像分割可视为将 R 分割成 n 个子区域 $R_1, R_2, \cdots, R_n$ 的过程，而这些子区域必须满足

(1) $\bigcup_{i=1}^{n} R_i = R$

(2) R_i 是相连接 (connected) 的区域，$i = 1, 2, \cdots, n$

(3) $R_i \cap R_j = \phi$ (空集合)，$\forall i, j,\quad i \neq j$

(4) $P(R_i) = \text{TURE}$，$i = 1, 2, \cdots, n$

(5) $P(R_i \bigcup R_j) = \text{FALSE},\quad i \neq j$

其中 $P(R_i) = \text{TURE}$ 代表在 R_i 中所有像素都有某种共同的特性，例如全部像素的灰度都相同。

以区域法实现分割便是遵循上述的概念，以直接找取区域的方式实现图像分割。主要方法有二，一是所谓的像素聚类区域成长法 (region growing by pixel aggregation) 以及区域分割与合并法 (region splitting and merging)。

- **像素聚类区域成长法**

顾名思义，此方法从一种子 (seed) 像素开始，通过像平均灰度、组织纹理及色彩等性质的检视、将具类似性质的像素逐一纳人所考虑的区域中，使其逐渐成长。此方法在概念上很简单，但在实际使用时必须考虑许多因素。

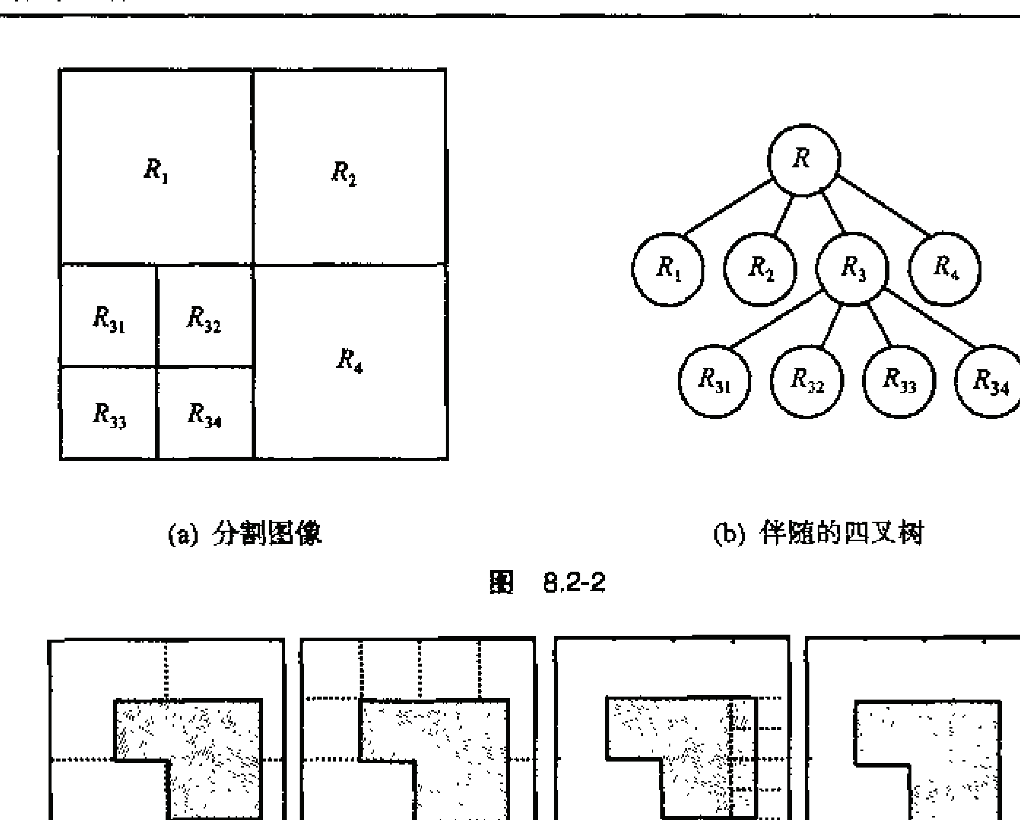
首先是种子像素的选择，这又分是否事先对所考虑的图像的特性已有了解。若已了解，则较易选定，例如在红外线图像的军事用途上，通常被追踪的目标温度较高，因此在红外线图像上亮度较高。若无足够信息直接选取种子像素，可对图像进行直方图分析，找到出现次数较大者当做种子像素。

另一个问题是聚类的相似性的选择。这个问题与应用对象有密切关联性，不过大致上不外乎灰度、组织纹理、色彩等性质。此外，何时该停止成长也必须考虑，当然最简单的就是说再也找不到符合聚类性质的像素时就该停止。其他的停止条件还可包括区域大小及形状等。

在实际的应用中，也许无法一次就找到满意的分割，因此有一些迭代式的方法。首先判别分割边界线两边的边界点性质的差异是否很大，若是则称为强 (strong) 边界、反之则为弱 (weak) 边界。强边界保留不动，弱边界可去除，使部分边界合并。此过程持续进行到没有一个弱边界为止。

- **区域分割与合并法**

首先将图像分割成不重叠的区域，其中最常用的方法是分割成四个大小相同的子图像。若每个子图像内有性质不同的图像存在就将该子图像继续分割，持续这个过程直到没有可再分割的条件为止。接着将具同性质又相邻的子区域进行合并，直到无法再合并为止，整个图像分割程序才算大功告成。图 8.2-2 显示一个图像分割情形及其伴随的四叉树 (quadtree) 数据结构表示方式。图 8.2-3 则显示用区域分割与合并的例子。



(a) 分割图像　　　　(b) 伴随的四叉树

图 8.2-2

图 8.2-3 区域分割与合并法的例子

8.2-3 边界法实现图像分割

边界法是借助求一幅图像的梯度大小来正确地找出边界的图像分割法。设一输入图像为 $f(x,y)$，则其梯度为向量

$$\nabla f(x,y)=\frac{\partial}{\partial x}f(x,y)\boldsymbol{i}+\frac{\partial}{\partial y}f(x,y)\boldsymbol{j} \tag{8.2-1}$$

其大小 $|\nabla f(x,y)|$ 代表在梯度向量所指的方向上 $f(x,y)$ 每单位距离的最大增加率。求此梯度大小的目的是要找出图像中灰度变化最大的位置，而这些灰度变化很大的位置，通常正好就是物体的边界。

- **边界追踪**

首先我们对一幅图像求一阶导数得到一幅梯度大小的图像。由梯度大小的图像找出一些灰度值较大的像素，从这些像素中再挑一个出来(例如最大者)作为追踪的起点，接着在与起点相邻的点中找出灰度最大的点当做第二点。现在把起点当做过去的边界点，而第二点当做现在的点，再从与现在的边界点相邻的点中，找出位于与过去的边界点相对面的像素，作为下一个候选边界点，从中选择灰度较高者作为下一个边界点，如图 8.2-4 所示。如果三个或相邻两个候选边界点有相同的最大灰度值，则选择中间者为下一个边界点；如果不相邻的两

个候选边界点有相同的最大灰度值，则此两点中任一点都可选为下一个边界点。如此重复，最后找出整个边界。

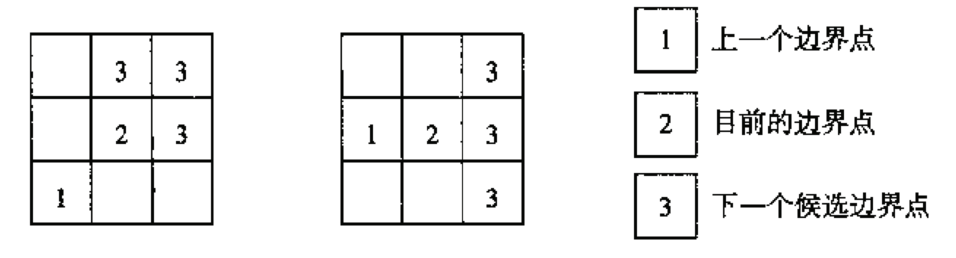


图 8.2-4　边界追踪

边界追踪法适合在没有干扰的情况下操作；假设梯度图像中边界已被干扰破坏，则显然极易追踪错误。其解决之道有二：一是在追踪前先做平滑 (smoothing) 去噪声，二就是下面要提到的追踪虫 (tracking bug) 的方法。

首先我们先定义矩形大小作为检视的窗口，如图 8.2-5，我们称之为虫。在图上可以看到有一个过去的边界点及一个现在的边界点，我们选择以由过去及现在的边界点所构成的方向，作为目前的方向。以此方向为准的 $\pm\theta$ 方向内都是下一个边界点的搜寻范围，接着计算在搜寻范围内各个位置的虫所涵盖的点的梯度平均值，选择平均值较大者所含的候选边界为下一个边界点。这种方式不同于上述之处是在于：此法所用的窗口拉大了搜寻有较高灰度点的范围，而这个窗口的大小，可依实际的情形而调整。因为整个方法的动作犹如虫一步一步往前进，故得其名。

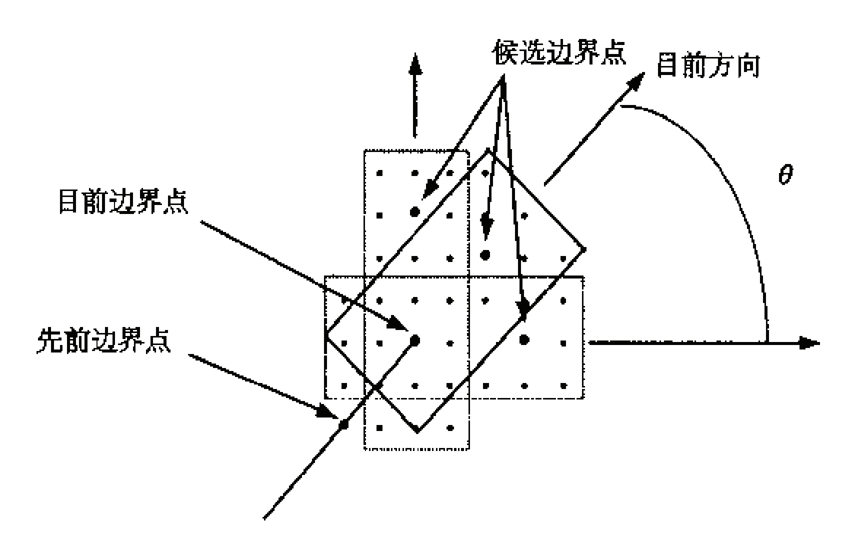


图 8.2-5　边界追踪虫

- **梯度大小阈值法**

图 8.2-6 显示梯度大小图像某一行的灰度值变化。一开始本方法从较小的阈值开始 (例如图 8.2-6 中的 T_1)，此时有许多相连的部分超过此阈值，接着逐渐增加阈值时，可相连的部分逐渐变小，到阈值等于 T_2 时发现相连部分只剩一点，该点即视为边界点，同理当阈值逼近 T_3 时又得另一边界点，依此类推。

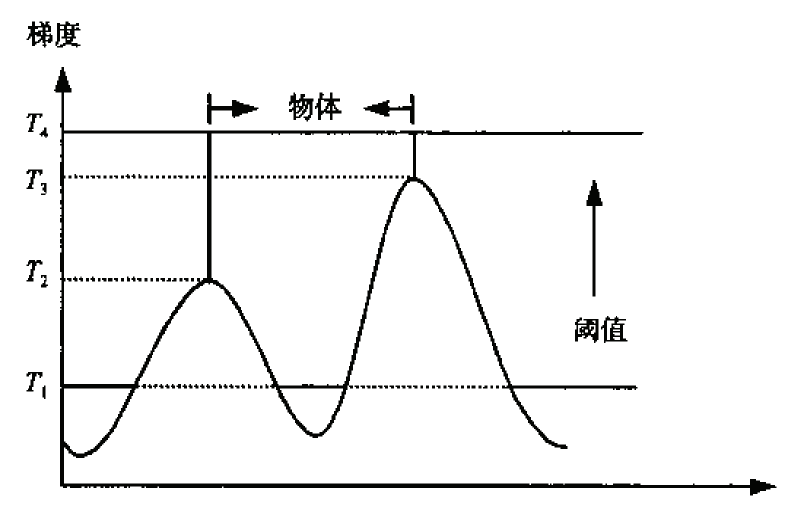


图 8.2-6　梯度大小阈值分割法的示意图

这样的作法有一个很大的好处，就是在一幅有多个物体及背景变化丰富的图像上，可以找出较正确的边界。但图像在做梯度运算前，须先平滑以降低噪声对边界的损害。

● **拉氏边界检测法**

二阶导数也是一个搜寻边界不错的方法。*拉氏边界检测* (Laplacian edge detection) 用的是二阶导数的方法，其定义为

$$\nabla^2 f(x,y)=\frac{\partial^2}{\partial x^2}f(x,y)+\frac{\partial^2}{\partial y^2}f(x,y) \tag{8.2-2}$$

在数字图像上是使用图 8.2-7 所示的*拉氏卷积核* (Laplacian convolution kernels) 中的一个去做二阶导数的计算。图 8.2-8 中可看到一边界的一阶导数及二阶导数的表示图形。一阶导数是以最大值作为边界的设定，而二阶导数是以其陡峭下滑且*越零点* (zero-crossing) 的位置来找寻边界。

0	-1	0
-1	4	-1
0	-1	0

-1	-1	-1
-1	8	-1
-1	-1	-1

图 8.2-7　拉式卷积核

由于 Laplacian 算子是二阶导数，因此它对于噪声有极高的敏感性，而且对于双边缘带不易检测出边缘的方向。基于这些原因，Laplacian 较一阶导数少用。之前谈到过要降低噪声的影响，可先做平滑操作，而具有 Gaussian 脉冲响应的低通滤波器便是一个很好的选择。设 Gaussian 形式的脉冲响应为 $h(x,y)=\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$，其中 σ 为标准差。则取 Gaussian 响应的 Laplacian，就成了拉氏—高斯滤波器的响应

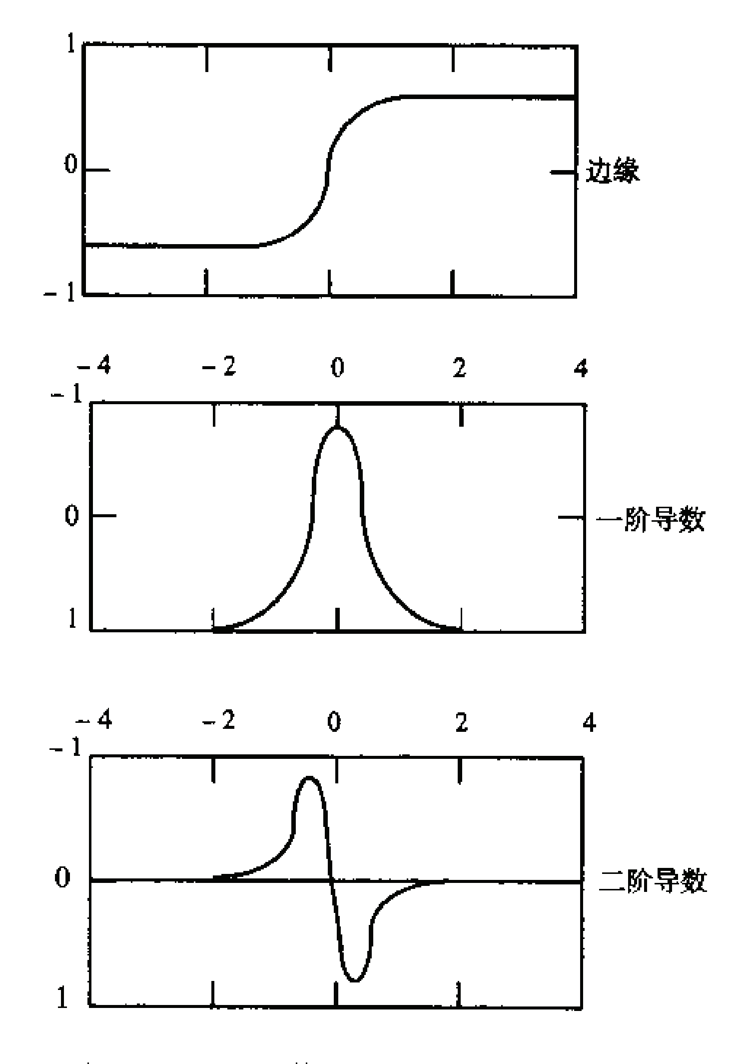


图 8.2-8　一边界的一阶导数及二阶导数的表示图形

$$\nabla^2 h=\left(\frac{r^2-\sigma^2}{\sigma^4}\right)\exp\left(-\frac{r^2}{2\sigma^2}\right) \tag{8.2-3}$$

其中$r^2=x^2+y^2$。图 8.2-9 是拉氏—高斯滤波器的响应图。

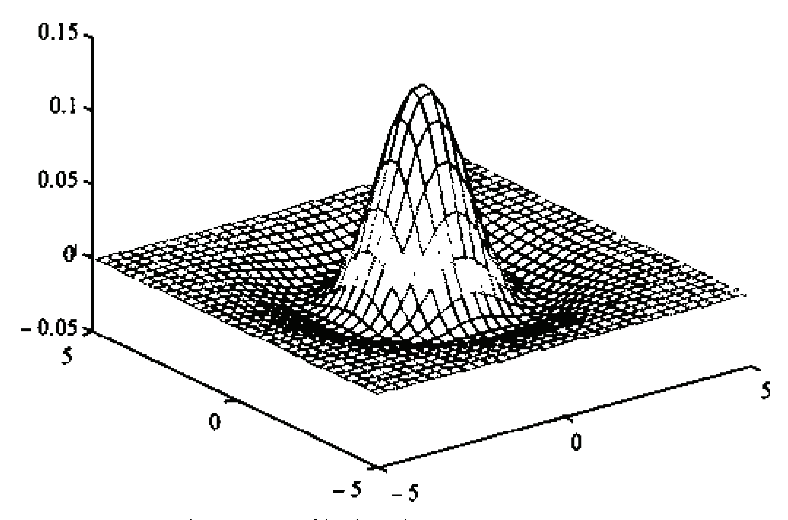


图 8.2-9　拉氏—高斯滤波器的响应图

8.2-4 边缘法实现图像分割

边缘法是另一种建立边界的方法，它是用一阶导数的大小检测出边缘 (edge) 所在并用一阶导数的方向将小的边缘连结成边界的方法。

● **检测** (Detection)

在这里“检测”是指利用一阶导数算子来对图像求梯度，以找出边缘。在数字图像处理中，求梯度通常是利用下列几种算子来实现：

Sobel 边缘算子 (垂直与水平方向的边缘)

-1	-2	-1
0	0	0
1	2	1

-1	0	1
-2	0	2
-1	0	1

Prewitt 边缘算子 (垂直与水平方向的边缘)

-1	-1	-1
0	0	0
1	1	1

1	0	-1
1	0	-1
1	0	-1

The Kirsch 边缘算子 (8 个方向的边缘)

5	5	5
-3	0	-3
-3	-3	-3

-3	5	5
-3	0	5
-3	-3	-3

-3	-3	5
-3	0	5
-3	-3	5

-3	-3	-3
-3	0	5
-3	5	5

-3	-3	-3
-3	0	-3
5	5	5

-3	-3	-3
5	0	-3
5	5	-3

5	-3	-3
5	0	-3
5	-3	-3

5	5	-3
5	0	-3
-3	-3	-3

● **连结** (Linking)

在理想状况下，由前面检测出的边缘点图像所形成物体的边界会是相连且封闭的。但实际上由于噪声、光源不均匀等因素通常得到的结果是残缺不全的边界，所以要作连结修补。

最简单的连结方法是用一个小视窗 (3×3 或 5×5)，将具有某种共同特性的边缘结合，此特性可包括梯度大小及方向。在一个阈值内具有相近的梯度大小及方向者将其连结成边界。相对这个以小视窗为考虑范围的局部 (local) 处理，接下来将介绍一个整体 (global) 处理做连

结的方法——Hough 变换。

我们由图 8.2-10 来解释 Hough 变换。假设在x,y的坐标上有几个点，如图 8.2-10 (a) 所示。由它们分布的情况来看，可完全由一直线方程式来代表

$$y = ax + b \tag{8.2-4}$$

其中 a 与 b 为常数。由于仅知道几个点的位置(x,y)，我们将其转换至a,b坐标上

$$b = y - ax \tag{8.2-5}$$

现在视 x 与 y 为常数。从几个不同的点可画出几条线来，如图 8.2-10(b) 所示。假设现在有两条线$b = y_1 - ax_1$与$b = y_2 - ax_2$相交于(a,b)的点上，这就代表了此二点(x_1,y_1)与(x_2,y_2)有共同的方向，再把它们相连。当然在x,y坐标上分布的点，并非刚好很准确地可用一方程式表示，所以我们将a,b平面做一个量化，也就是分为好几个小区域来表示；若解出的(a,b)在同一区域中，则称其同一方向，并做一统计：各区域伴随一累加器，若解出在某区时，该区域的累加器就加 1。依次做完所考虑的点后，就可以看到它们大致分布的情形，也可以找出一条大致可代表这些点的曲线。我们可将x,y坐标表示为极坐标，依同样的观念，来做 Hough 变换，如图 8.2-10 (c) 所示。

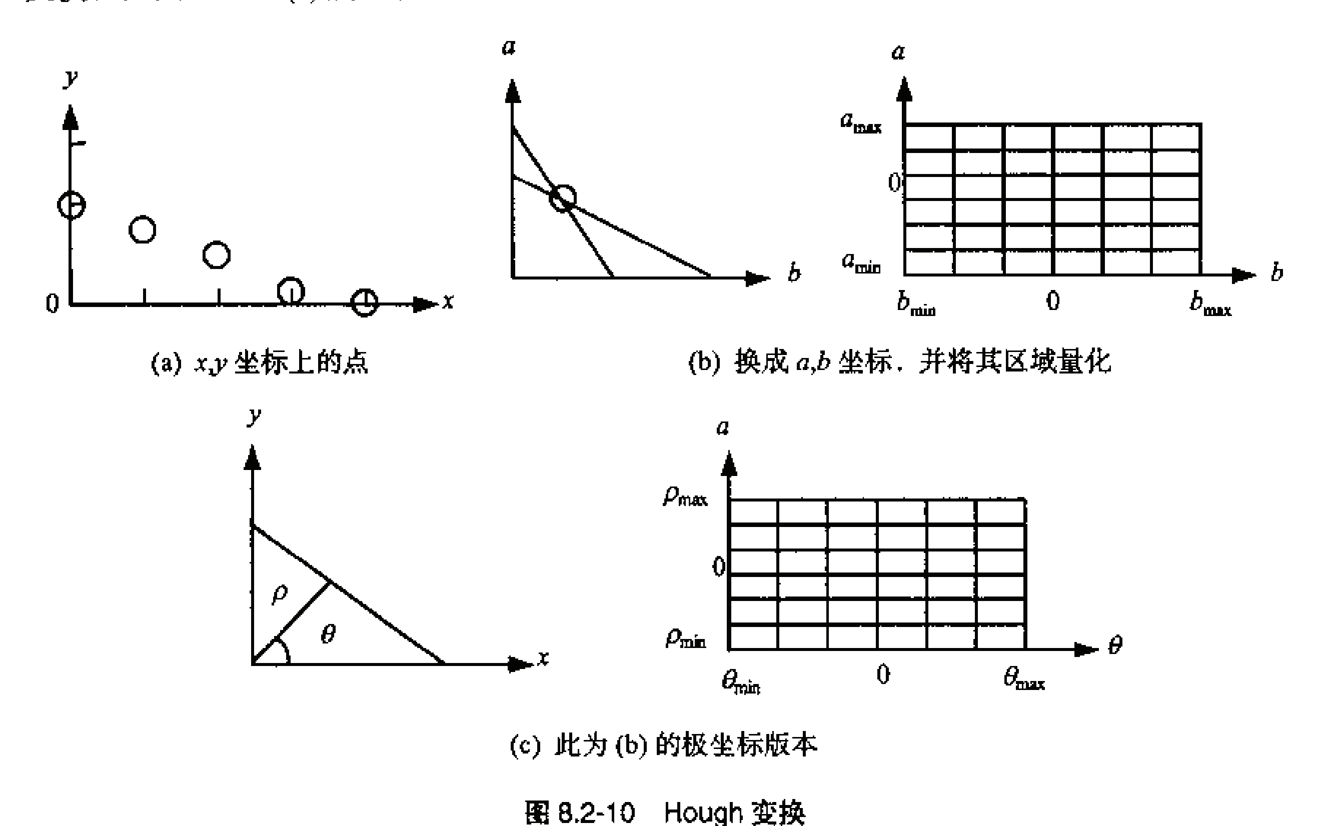


图 8.2-10　Hough 变换

8.3　分割图像的储存

在图像分析中，我们常希望将物体重新排列、整理，或是需要将物体个别展示，所以我

们必须将物体抽出并且以更方便的形式储存起来。以下介绍两种方式，分别是边界连锁及线段编码法。

8.3-1 边界连锁码

边界连锁编码 (boundary chain code) 是以边界来定义物体，所以它不必储存物体内部像素的位置。假设我们从一位于 x, y 平面的物体的边界出发，在边界上任选一点，而在以它为中心与其相邻的点中一定也存在有边界点。现在这八个相邻点有八个不同的方向，分别以 0 ~ 7 代表这八个方向，如图 8.3-1 所示。

3	2	1
4		0
5	6	7

图 8.3-1 边界追踪方向

如此以边界追踪的方式寻找边界，并将每个边界的位置储存起来，便可将物体的数量、周长以及连锁编码记录在存储器中。这种方法最大的好处就是节省存储器空间，但由于物体内部点并没有储存，故这方面的数据并没有留下，不过若我们所感兴趣的只是边界，那么内部的数据记录与否就没有关系了。

8.3-2 线段编码

线段编码 (line segment encoding) 与上述的方法不同的地方就是它可储存物体中每一点的数据。这是一种以线段储存的方法。我们以图 8.3-2 为例来说明其储存的操作。

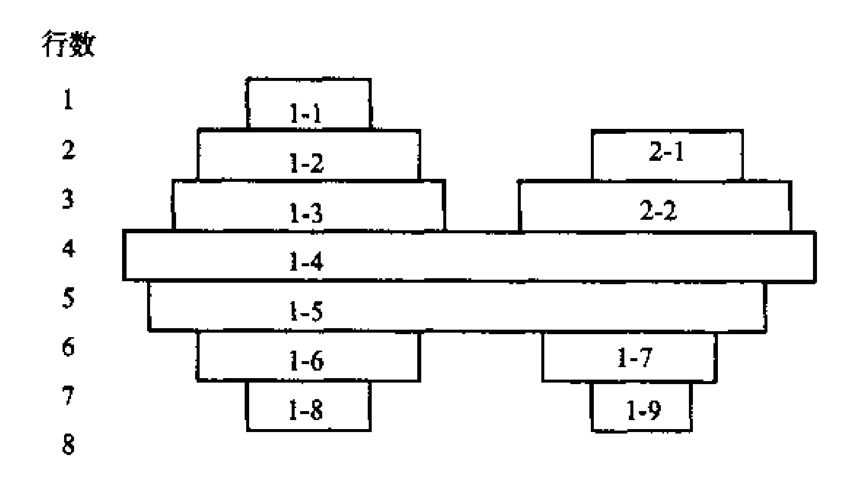


图 8.3-2 物体的线形片段表示

假设现有物体如图 8.3-2 所示。在第 1 行时检测到有一列分割的图像，接着我们就把它视为第一个物体中的第一列，符号记下 1-1，接着在第 2 行检测到有两列；第一列因处于 1-1

的下方，所以记做 1-2；而第二列为一新的物体，所以记做 2-1。如此检测到第 4 行时发现只有一列且位于物体 1 及物体 2 的下方，所以原先视为两个物体的图像原来为同一物体，但先记做 1-4，等待全部扫描完之后，再作合并的操作。每个物体储存的信息包括有：第几个物体、共分为几列，接着是各列的数据。在此例中因物体 1 和物体 2 是同一物体，但在全部未扫描完之前以此储存，待扫描完后，物体 2 将与物体 1 合并记录下来。

这种技巧所记录下来的数据可包括：区域面积、周长、物体特征、分割的图像的大小、宽度以及物体的总数等；较前述方法储存的数据多了许多，但相对也要占用较大的存储器空间。

在本章一开始时提到去除噪声与去除模糊等工作是属于图像分析的前期处理，它是方便像图像分割这种中层处理而设计。虽然在第四章图像增强与第五章图像恢复中提到这些底层工作，但是以下将介绍一种更强而有力的方法，它是同时可去除噪声及模糊的图像分割预处理方法，对实际上面对的图像分割问题非常有帮助。

8.4 图像分割预处理——LUM 滤波器

本节将介绍图像分割预处理所使用的 LUM (lower-upper-middle) 滤波器。LUM 滤波器名称的由来乃是因为在滤波器的演算法则中是以样本中较低阶统计量 (lower order statistics) 及较高阶统计量 (upper order statistics) 与样本中间值做比较以运算出输出值。借助适当的参数选取，LUM 滤波器可以当做平滑化滤波器、亦可当做锐化滤波器使用。

以排序为基础的滤波器被广泛地运用来达到平滑化效果。首先便是到目前为止可能仍是最广为被运用的中值滤波器，中值滤波器及其特性在本书 5.2 节中已有详细描述，此处不多做赘述。在许多应用中，中值滤波器会引入过多的平滑效果，而这些被引入的模糊效果可能会对后续的处理产生比图像原始噪声 (original noise) 更不利的影响。在 LUM 滤波器中，平滑特性由 2 个参数中的第 1 个参数所控制，这个参数的调整可在从无平滑效果到与中值滤波等效的区间内改变平滑化级别。借助控制这个参数可在噪声平滑化与保存图像细节之间得到一个良好的平衡。

LUM 滤波器亦可被设计成具有增强边缘特性的功用，而增强的幅度可借助控制 LUM 滤波器的第 2 个参数来达到。传统上边缘的增强与锐化都是使用线性的技巧来达到，包括 Wiener 滤波器与高通滤波等。在许多情况下，线性技巧均有良好表现，但也有许多不理想的情况，例如线性锐化器经常导致边缘增强得太过与不足，同时也具有放大背景噪声的倾向。相形之下，LUM 滤波器既不会有上述情况，又可在增强边缘特性的同时达到对相加性噪声不敏感及去除脉冲型噪声的效果。

由于 LUM 滤波器可避免传统平滑滤波器与线性锐化器的缺点，故 Hardie 与 Boncelet 曾提出以 LUM 滤波器为预滤波器的 Sobel 算子边缘检测方法，并证实其效果比以中值滤波器为预滤波器时良好。总之，LUM 滤波器有如下的优点：

① 不须牺牲其简易性便具有多用途功能。

② 使参数的选择变得单纯且直觉。

③ 同时具有去除噪声与保存信号特性的良好效果。

④ 可根据图像特性调整其性能。

● **LUM 平滑滤波器**

考虑一含有 N 个样本的窗函数 (window function)，如图 8.4-1 所示。其中窗函数的中心样本为 x^*，N 为奇数。其表示式为

$$W=\left\{x_1,x_2,\cdots,x_N\right\} \tag{8.4-1}$$

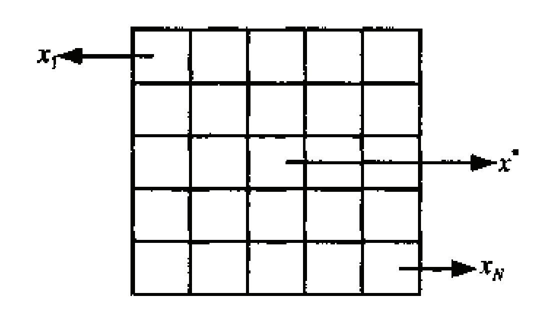


图 8.4-1　含有 N 个样本的窗函数

此样本集的排序形式可写为

$$x_{(1)}\leqslant x_{(2)}\leqslant\cdots\leqslant x_{(N)} \tag{8.4-2}$$

定义 1：具有 k 参数的 LUM 平滑滤波器其输出为

$$y^*=\operatorname{med}\left\{x_{(k)},x^*,x_{(N-k+1)}\right\} \tag{8.4-3}$$

此处 y^* 为窗函数中心样本的估测，$x_{(k)}$ 为样本集中的较低阶统计量，$x_{(N-k+1)}$ 为样本集中的较高阶统计量，$1\leqslant k\leqslant(N+1)/2$。

根据 (8.4-3) 式可知，当 $x^*<x_{(k)}$ 时，LUM 平滑滤波器输出为 $x_{(k)}$。若 $x^*>x_{(N-k+1)}$ 时，输出为 $x_{(N-k+1)}$，否则输出即为 x^*，如图8.4-2所示。

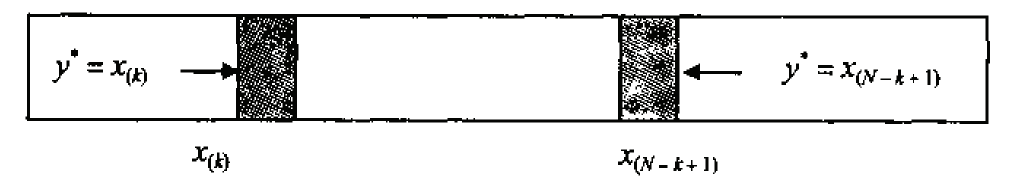


图 8.4-2　LUM 平滑滤波器的输出

在定义中将中间样本与较低阶及较高阶统计量做比较的理由是：这些排序的统计量形成一个常态值化 (normal-valued) 的样本范围。若 x^* 位于范围内，则将不对其做任何修正；反之如果 x^* 位于范围外，则以较接近样本集中间值的样本取代，因此这样的方式具有平滑的功能。例如：若 x^* 为一脉冲型噪声，则显然它极可能落在低阶统计量及高阶统计量所形成的范

围之外，则x^*将被较接近样本集中间值的样本取代、而噪声也随之移除。

参数k控制了滤波器的平滑特性，并且可经调整以使平滑噪声与保存信号细节间的最佳平衡。当$k=(N+1)/2$时，LUM 平滑滤波器的输出为W的中值，同时这也是 LUM 平滑滤波器输出所能达到的最大平滑效果。当k递减时，滤波器的细节保存特性亦随之改善。当k为1时，LUM 平滑滤波器成为一个恒等滤波器 (identity filter) (亦即$y^*=x^*$)。

● **LUM 锐化滤波器**

LUM 平滑滤波器与其他以排序为基础的滤波器一样均是借助将样本朝中值方向移动来得到平滑特性。但是为了得到锐化特性，样本必须是朝向极端不同的排序方向移动，换言之，也就是远离样本集的中间值。这同时也是 LUM 锐化滤波器的运算原理。

在定义 LUM 锐化滤波器之前，我们必须先定义一个位于低阶统计量与高阶统计量($x_{(l)}$与$x_{(N-l+1)}$)中间的值。这个中间点 (或是平均值)，表示成t_l，其定义如下

$$t_l=\frac{1}{2}\left(x_{(l)}+x_{(N-l+1)}\right) \tag{8.4-4}$$

此处$1\leqslant l\leqslant (N+1)/2$。

定义 2：具有l参数的 LUM 锐化滤波器其输出为

$$y^*=\begin{cases}x_{(l)}, & \text{如果}x_{(l)}<x^*\leqslant t_l\\ x_{(N-l+1)}, & \text{如果}t_l<x^*<x_{(N-l+1)}\\ x^*, & \text{其他情况}\end{cases} \tag{8.4-5}$$

如此一来若$x_{(l)}<x^*<x_{(N-l+1)}$，则x^*会根据与$x_{(l)}$或$x_{(N-l+1)}$间何者距离较近、而向外远离该点、否则x^*将不会有任何改变，如图 8.4-3 所示。这个锐化操作的理由是、若$x_{(l)}<x^*<x_{(N-l+1)}$，则x^*可视为一斜坡上的一个转折点 (transition) 样本。借助将x^*往两个极端不同方向移动，我们移去了转折点样本因而创造一个更陡峭的落差。

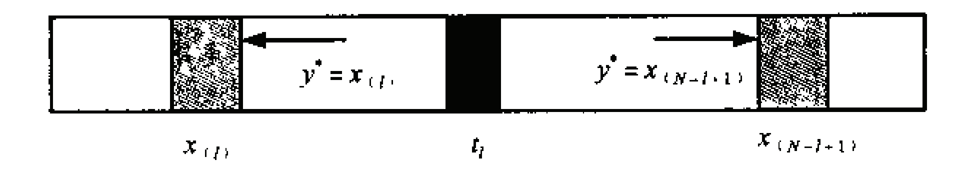


图 8.4-3　LUM 锐化滤波器

借助改变参数l的值可达到使锐化级别变化的目的。在$l=(N+1)/2$的情况下，没有任何锐化效果，此时 LUM 锐化滤波器仅是一恒等滤波器。当在$l=1$的情况下，由于x^*往两个极端方向统计量$x_{(1)}$或$x_{(N)}$移动的缘故，会造成最大的锐化效果。

● **LUM 滤波器**

为了得到一个效能良好且又能滤除噪声的加强滤波器，必须将 LUM 平滑滤波器及锐化滤波器的概念加以结合，如此就得到一个通用的 LUM 滤波器。在定义 LUM 滤波器前，我们须先定义如下的低阶与高阶统计量

$$x^L = \text{med}\left\{x_{(k)}, x^*, x_l\right\} \tag{8.4-6}$$

$$x^U = \text{med}\left\{x_{(N-k+1)}, x^*, x_{(N-l+1)}\right\} \tag{8.4-7}$$

此处$1 \leqslant k \leqslant l \leqslant (N+1)/2$，注意其中$x^L \leqslant x^U$。

通用 LUM 滤波器的输出由x^U或x^L二者间较靠近中心样本x^*者给定。参数k及l可视为可调整型参数，将使得 LUM 滤波器的特性具有变化性。稍后的叙述中将有更多的说明。

下面将列出两组等义但形式略有不同的数学等式，亦即 LUM 滤波器的定义。其中定义 3 (b) 较为直接。

定义 3 (a)：LUM 滤波器的输出为

$$y^* = \begin{cases} x^L, & \text{如果 } x^* \leqslant \left(x^L + x^U\right)/2 \\ x^U, & \text{其他情况} \end{cases} \tag{8.4-8}$$

定义 3 (b)：另一个 LUM 滤波器的定义为

$$y^* = \begin{cases} x_{(k)}, & \text{如果 } x^* < x_{(k)} \\ x_{(l)}, & \text{如果 } x_{(l)} < x^* \leqslant t_l \\ x_{(N-l+1)}, & \text{如果 } t_l < x^* < x_{(N-l+1)} \\ x_{(N-k+1)}, & \text{如果 } x_{(N-k+1)} < x^* \\ x^*, & \text{其他情况} \end{cases} \tag{8.4-9}$$

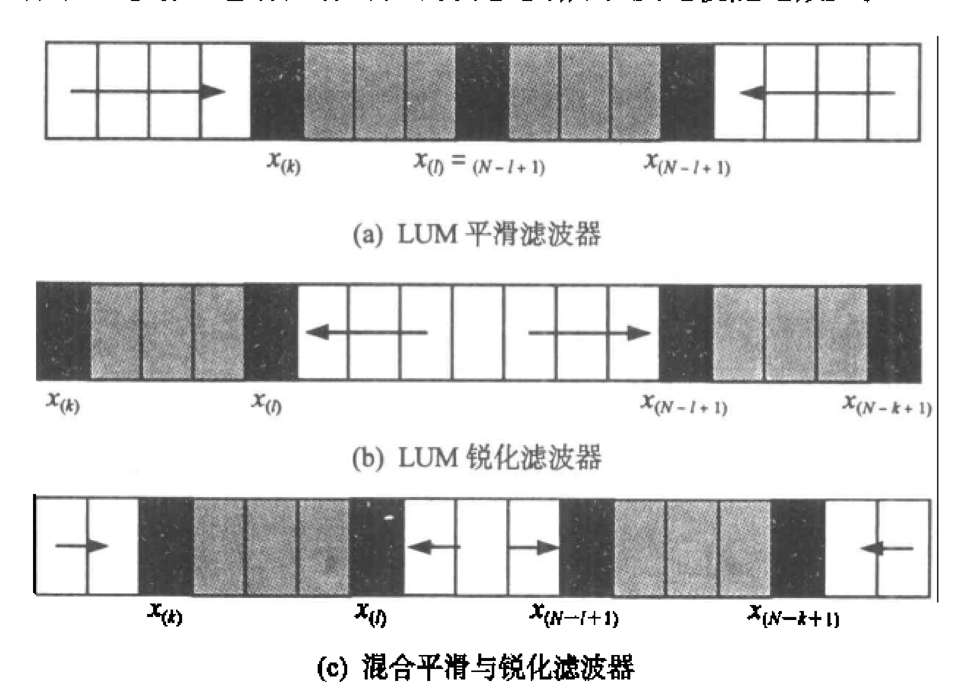
此处t_l为在 (8.4-4) 式中所定义的中间点。

不考虑如何实现的问题，LUM 滤波器的表现展现了关联性简单的功能。若$x^* < x_{(k)}$时，LUM 滤波器的输出为$x_{(k)}$；同理，若$x^* > x_{(N-k+1)}$时，LUM 滤波器的输出为$x_{(N-k+1)}$。如此一来，便如 LUM 平滑滤波器一般，样本$x_{(k)}$与$x_{(N-k+1)}$在 LUM 滤波器的输出上形成了界限。另一方面、若$x_{(l)} < x^* < x_{(N-l+1)}$时，则中心样本会根据$x^*$与$x_{(l)}$或$x_{(N-l+1)}$间何者距离较近、而向外远离该点，就如同 LUM 锐化滤波器一般。若x^*的位置是在$x_{(k)} \leqslant x^* \leqslant x_{(l)}$或$x_{(N-l+1)} \leqslant x^* \leqslant x_{(N-k+1)}$时，则滤波对该样本无任何影响。

借助操控参数k与l，LUM 滤波器呈现了其特性的变化。为了描述这样的变化及阐明 LUM 滤波器的定义，图 8.4-4 描述了一些特例及通例下的滤波程序。图中呈现了一组排序统计量的集合。样本范围自左至右的变化为由小至大递增。阴影的部分表示中心样本不受影响的范围。

图 8.4-4 (a) 显示当$l = (N+1)/2$且k值变动的情况。此处滤波器的功能为 LUM 平滑滤波器。当k值递增时，其平滑程度亦随之增加。若$k = l = (N+1)/2$，则此时 LUM 滤波器相当于中值滤波器。图 8.4-4 (b) 描述当$k = 1$且l值变动的状况，在此情况下我们得到一 LUM 锐化滤波器。当l值递减，$x_{(N-l+1)}$与$x_{(l)}$值分别趋向两端。这样的现象导致一递增的增强效应。图 8.4-4 (c) 显示当$1 \leqslant k \leqslant l \leqslant (N+1)/2$ 的情况。在这个状况下，可以同时达到锐化与滤除噪

声的效果。参数 k 递增以增进 LUM 滤波器去除脉冲噪声的特性，另一方面，参数 l 控制着增强效果的级别，亦即参数 l 递增，增强效果亦随之增加，反之便随之减少。



(c) 混合平滑与锐化滤波器

图 8.4-4　LUM 滤波器的运作

此处以一区域性实例来说明 LUM 滤波器的平滑及锐化作用。设有一10×10 的图像区块，其灰度值分布如图 8.4-5 (a) 所示。图 8.4-5 (a) 中以方框圈起的像素为所加入的脉冲型噪声，线段则为边缘所在位置。将此区块用一个5×5 的模板滑动遮盖，参数 $k=5$ 及 $l=6$ 的 LUM 波器运算后，其结果如图 8.4-5 (b) 所示。由图 8.4-5 (b) 中可看出，原本存在的脉冲型噪声经运算后已被平滑化，同时边缘特性也被加强，亦即边缘所在位置的像素的灰度值差距被扩大。

159	159	153	54	56	57	221	55	58	56
160	158	153	56	55	54	55	56	53	56
45	159	153	56	56	52	54	58	222	53
150	165	153	56	55	54	56	58	57	53
153	160	165	153	56	56	50	53	52	58
153	150	154	162	153	56	51	54	53	55
154	155	155	167	153	56	51	200	55	56
154	155	155	159	153	59	53	56	56	56
154	155	10	159	153	55	49	53	53	53
154	155	152	158	153	56	56	56	56	56

(a) 运算前的 10×10 图像区块

图 8.4-5　LUM 滤波器的区域运算实例 (待续)

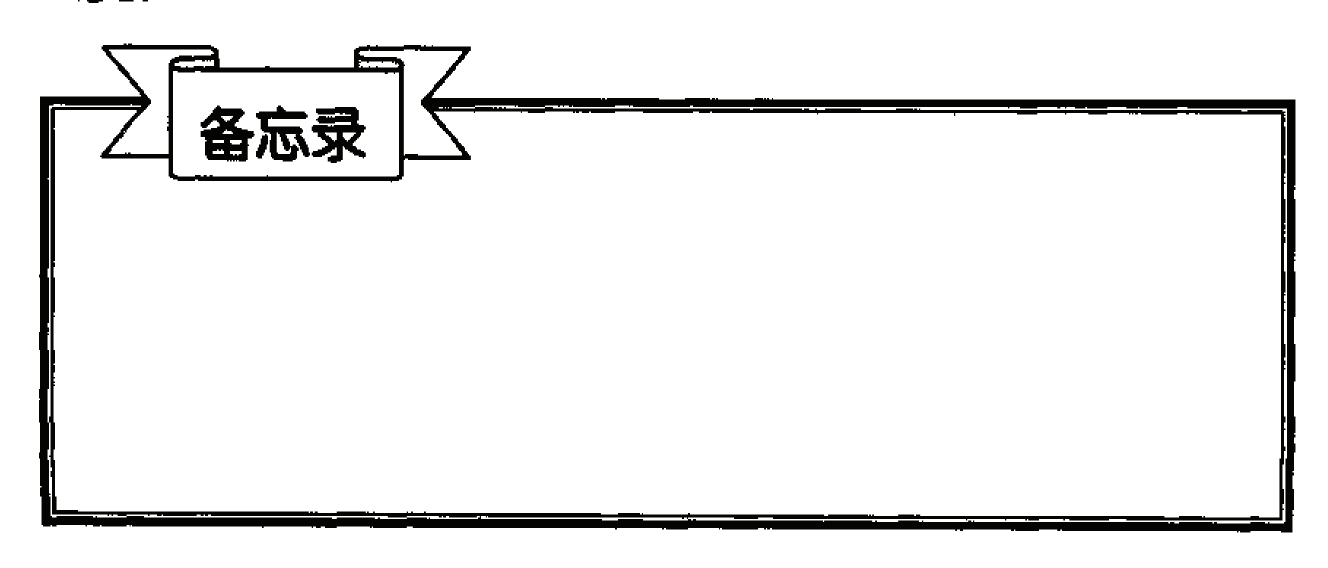
159	159	159	54	54	57	58	54	58	58
160	159	159	55	54	54	54	57	53	58
153	160	159	55	54	54	54	58	58	53
150	160	159	56	54	54	56	57	57	53
150	160	160	159	54	54	52	53	52	58
153	153	153	160	155	54	53	53	53	53
153	153	153	159	155	53	52	56	56	56
153	155	155	158	155	54	53	56	56	56
154	155	153	158	155	53	53	53	53	53
154	155	153	158	155	55	53	56	56	56

(b) 经运算后的 10×10 图像区块

图 8.4-5(续)

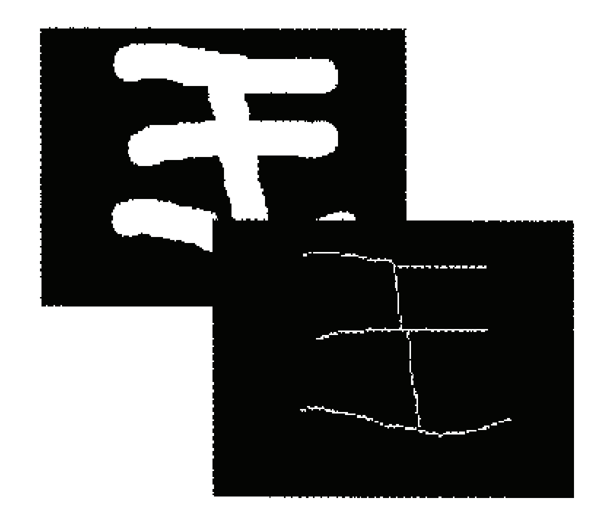
习　题

1. 考虑一个二值 (binary) 图像，其中 1 代表物体、0 代表背景。试提出二套算法，分别用来决定其四连通及八连通物体。
2. 考虑图 8.4-5 所示的图像，如果要以像素聚类成长法来执行分割，则像素种子、相似性及停止条件该如何来选取？
3. 将图 8.4-5 (b) 去掉外围的二行及二列以形成 8×8 的图像矩阵，对此新图像，重复上一题的问题。
4. 试以边界追踪法找出上一题的图像的边界。
5. 重复上题但改以梯度大小阈值法。
6. 试以拉氏边界检测法找出图 8.4-5 (a) 的图像的边界。
7. 重复上题但先以适当的 LUM 滤波器处理后再进行边界的寻找。与上一题的结果比较并讨论之。



第九章

表示与描述



图像分割形成区域之后，所形成的分割像素的集合通常是用适合于计算机进一步处理的形式给予表示和描述的。通常表示一个区域有两种方式：① 利用它的外部特性 (它的边界) 表示区域；或者② 利用它的内部特性 (组成区域的像素的性质) 来表示。选定一种表示方式后，接下来就根据所选择的表示来描述区域。例如一个区域可以用它的边界来表示，而边界是用诸如它的长度这一类的特征去描述。通常当所关切的重点是形状特征时，选择外部表示；当所关切的重点是如色彩和纹理等性质时，则选择内部表示。无论如何，作为描述子 (descriptor) 所选择的特征应该对大小、平移和旋转等变化不敏感，以方便以后自动识别。

9.1 表示方法

9.1-1 链 码

链码 (chain code) 是用一个由固定长度和方向的直线段所连接成的序列来表示一个边界的方法。这种表示的典型方法是以线段的 4 或 8 连通性为基础。每个线段的方向用如图 9.1-1 所示的数字方法来编码。

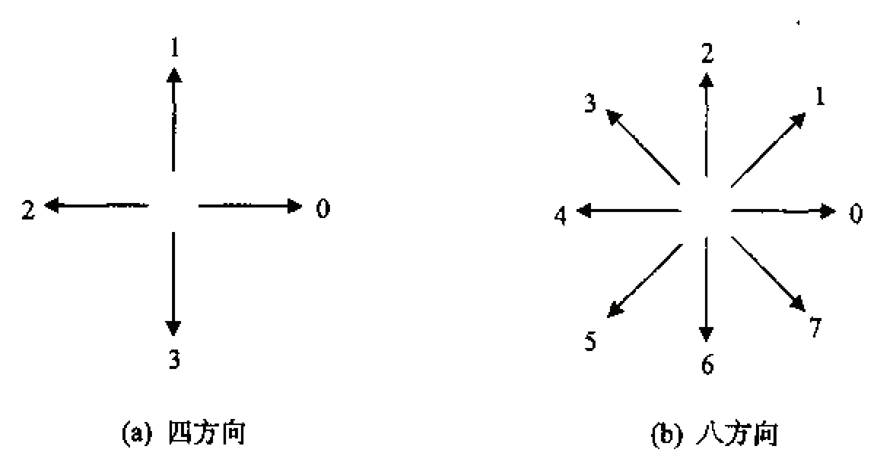
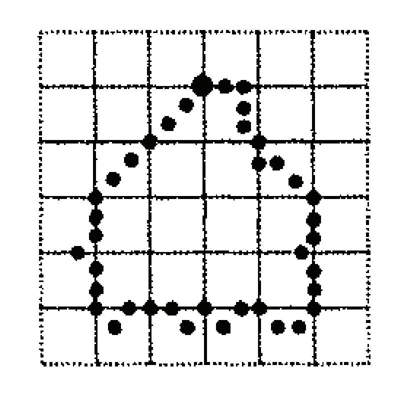


图 9.1-1 链 码

数字图像通常是用网格方式来表示和处理的，网格在 x 和 y 方向上等间隔。所以链码可以如此产生：沿边界给连接每一对像素的线段指定一个方向。图 9.1-2 (a) 显示一个链码产生的结果。首先从左上到右下的扫描顺序找到第一个边界点，接着按顺时针方向寻找下一个边界点，找到后就得一个链码，依此方式继续找出其他链码，直到回到第一个边界点时才停止链码的产生。

假使我们只需要比较粗略地描述边界即可，以节省储存空间或处理时间，则可做网格重新取样，再进行链码的产生，如图 9.1-2 (b) 所示。



链码：0066760776657766543453
534453223122111111

(a) 原始链码

链码：0676644442211

(b) 新链码

图 9.1-2　产生链码的例子

9.1-2　直线段近似

我们可用直线段近似一个物体的边界，其逼近程度可根据设定的临界值加以控制。有一个简单的直线段近似法如图 9.1-3 所示，其中 A 至 B 的曲线最后由 $\overline{AD}$ 、$\overline{DC}$ 、$\overline{CE}$ 及 $\overline{EB}$ 线段所近似。C、D 及 E 这三个折点是如何选取的呢？首先形成 $\overline{AB}$ 线段，则 C 点是离此线段距离最远者，若此距离超过某个临界值，则此 C 点为折点。设此 C 点确实为折点，则形成 $\overline{AC}$ 及 $\overline{BC}$ 线段，此时 D 与 E 点分别是与这两个线段距离最远者，凡是距离超过临界值者又再被设为折点，依此类推，直到没有任何距离超过临界值时才停止。若此曲线为封闭曲线，则 A 与 B 点可选择为曲线上相距最远的两点。

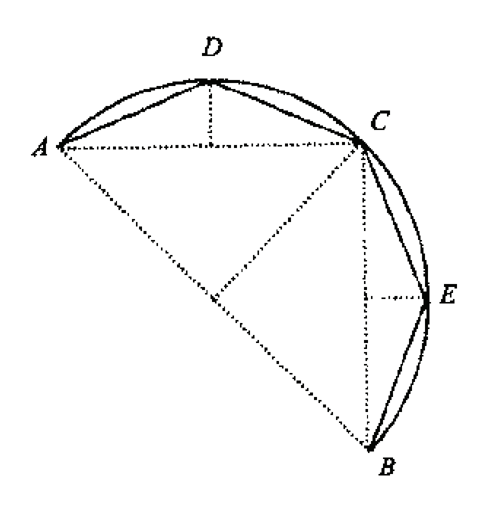


图 9.1-3　以直线段逐渐逼近曲线的示意图

我们可以直线段做曲线的逼近，自然也可以用抛物线或其他更高次的多项式函数来逼近，其目的是获得较平滑的近似结果。下一节介绍一种这样的逼近。

9.1-3 B-样条曲线拟合 (B-Spline Curve Fitting)

有时工程师常用称为样条 (spline) 的细杆，把经过各已知点的平滑曲线拟合在一起。因此样条一词亦适用于数学上的曲线拟合问题。

设一曲线可表成某一函数$f(x)$，其中$a \leqslant x \leqslant b$。假设$[a,b]$被下列各点分隔成子区间

$$a = x_0 < x_1 < x_2 \cdots < x_{n-1} < x_n = b \tag{9.1-1}$$

试在各子区间$[x_{j-1}, x_j]$上，以一多项式 $P(x)$ 来近似 $f(x)$。各子区间可以用不同的多项式，但必须把相连区间上所用的多项式拟合在一起，使所得的函数可微分二次。若所用的多项式不超过三次，则将各多项式拟合在一起所得的函数称为*三次样条函数* (cubic spline)。$f(x)$ 的三次样条逼近必须满足下列条件：

1. $$P(x_j) = f(x_j), \quad j = 0,1,2,\cdots,n \tag{9.1-2}$$

2. 在各区间$[x_{j-1}, x_j]$上，$P(x)$ 均有形式

$$P_j(x) = a_j + b_j x + c_j x^2 + d_j x^3, \quad j = 1,2,\cdots,n \tag{9.1-3}$$

3. $$P'(a) = f'(a), \quad P'(b) = f'(b) \tag{9.1-4}$$

4. 为了接合点的平滑性，需有

$$P_j(x_j) = P_{j+1}(x_j), \quad P_j'(x_j) = P_{j+1}'(x_j), \quad P_j''(x_j) = P_{j+1}''(x_j) \tag{9.1-5}$$

其中 $j = 0,1,2,\cdots,n-1$。

由条件 2 知共有 $4n$ 个未知数待求，而由其他条件可得$(n+1)+2+3(n-1)=4n$ 个线性独立的方程式，故$P(x)$ 可以有惟一解。如果有一接合处的位置有所改变，则 $P(x)$ 仍可有惟一解，但是此解可能与上一个解有很大的出入。换言之只要有一个接合点起变化，就会改变整个近似曲线的形状，因而难以作局部修正。此外，还有数值计算上不稳定的可能。因此在许多实际的应用中都用一种称为 B-样条 (B-spline) 的特殊函数以避免上述问题。

一个经过归一化的 k 阶 B-样条函数写成$B_{j,k}(x)$，其中 $j = 0,1,2,\cdots,n$; $k = 1,2,\cdots$。此函数可用下列迭代方式产生

$$B_{j,k}(x) = \frac{(x - x_j)B_{j,k-1}(x)}{x_{j+k-1} - x_j} + \frac{(x_{j+k} - x)B_{j+1,k-1}(x)}{x_{j+k} - x_{j+1}}, \quad k = 2,3,\cdots \tag{9.1-6a}$$

$$B_{j,1}(x) = \begin{cases} 1, & x_j \leqslant x < x_{j+1} \\ 0, & \text{其他} \end{cases} \tag{9.1-6b}$$

图 9.1-4 显示了一些 B-样条函数。这些函数有非负的值而且只在有限区间内 (finite support) 不为零。事实上，对于归一化的 B-样条，$0 \leqslant B_{j,k}(x) \leqslant 1$且不为零的区间为$[x_j, x_{j+k}]$。$B_{j,k}(x)$ 函数构成在逐段多项式函数的空间中的基底。参数 k 控制曲线连续或平滑的程度，例如 $k=3$，则样条是一个逐段的二次多项式；$k=4$ 则为三次多项式。

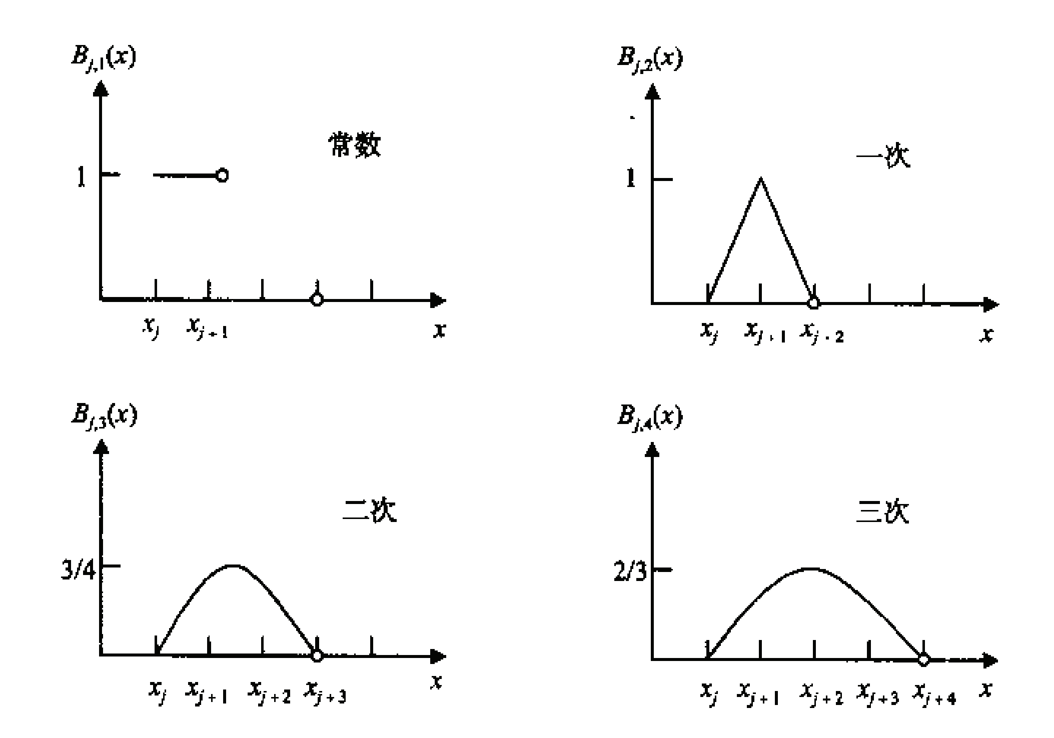


图 9.1-4　阶次为 k=1,2,3,4 的归一化 B-样条函数

设接合点相距为等间隔，即

$$x_{j+1}-x_j=\Delta x,\quad \forall j \tag{9.1-7a}$$

则所有子区间都有相同的基底函数，此时 $B_{j,k}(x)$ 可由 $B_{0,k}(x)$ 的平移获得，即

$$B_{j,k}(x)=B_{0,k}(x-j),\quad j=k-1,k,\cdots,n-k+1 \tag{9.1-7b}$$

如果 $\Delta x=1$ 而且选取 x_j 为正整数，则以 $k=3$ 为例可画出如图 9.1-5 所示的 $B_{j,3}(x)$。表 9.1-1 则列出 $B_{0,k}(x)$, $k=1,2,3,4$ 的函数。

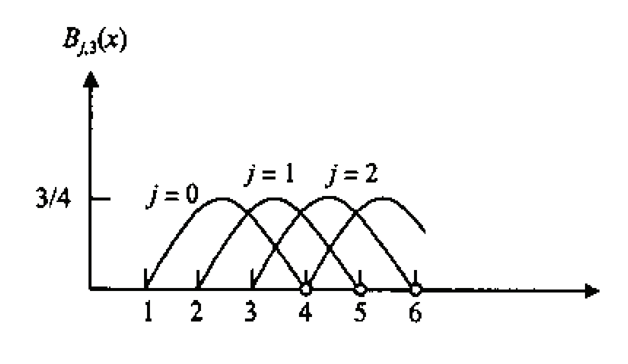


图 9.1-5　$B_{j,3}(x)$ 函数

由图 9.1-5 中可看出，在一个子区间内，例如$[x_2,x_3]=[3,4]$中，所用的拟合多项式通常是数个 (此处为 3 个) 样条函数的组合，即

$$P(x)=\alpha_0\beta_{0,3}(x)+\alpha_1\beta_{1,3}(x)+\alpha_2\beta_{2,3}(x) \tag{9.1-8}$$

其中 α_j，$j=0,1,2$，为待定的常数系数。

表 9.1-1　四个 B-样条函数

$B_{0,1}(x)=\begin{cases}1, & 0\leqslant x\leqslant 1\\ 0, & \text{其他}\end{cases}$	$B_{0,2}(x)=\begin{cases}x, & 0\leqslant x<1\\ 2-x, & 1\leqslant x<2\\ 0, & \text{其他}\end{cases}$
$B_{0,3}(x)=\begin{cases}\dfrac{x^2}{2}, & 0\leqslant x<1\\ -x^2+3x-1.5, & 1\leqslant x<2\\ \dfrac{1}{2}(3-x)^2, & 2\leqslant x<3\\ 0, & \text{其他}\end{cases}$	$B_{0,4}(x)=\begin{cases}\dfrac{x^3}{6}, & 0\leqslant x<1\\ \dfrac{1}{6}(-3x^3+12x^2-12x+4), & 1\leqslant x<2\\ \dfrac{1}{6}(3x^3-24x^2+60x-44), & 2\leqslant x<3\\ \dfrac{1}{6}(4-x)^3, & 3\leqslant x<4\\ 0, & \text{其他}\end{cases}$

由于 B-样条函数形成的函数 $P(x)$ 在每个子区间上最多由 k 个 B-样条函数来决定，而 B-样条基底函数本身只在 k 个子区间内不为零，因此有局部的特性。亦即改变像 (9.1-8) 式中的任何系数 α_i，只改变 k 个子区间曲线的形状。另外，三次的 B-样条函数也具备连续的一阶与二阶导数，因此所形成的样条曲线也非常平滑。

9.1-4　区域的骨架

一个平面区域的骨架 (skeleton) 可用来代表其形状结构，例如人及恐龙的骨架照片反应出两者截然不同的形状。

我们常用中轴变换 (medial axis transformation，MAT) 来定义一个区域的骨架。一个边界为 B 的区域 R 的 MAT 定义如下：对于在 R 中的每个点 P，寻找在 B 点中与其最近的点 (根据欧几里德或其他距离量测)，若最近的点在一个以上，则 P 属于 R 的中轴 (骨架)。图 9.1-6 显示以欧几里德距离为量测方式求一 MAT 的例子。

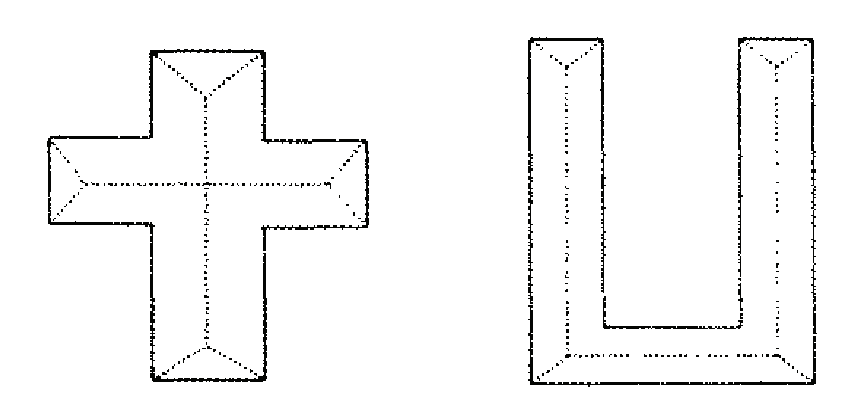


图 9.1-6　两个简单区域及其中轴 (以虚线表示)

直接用 MAT 的定义去找 MAT 需要付出极高的计算代价：考虑一个 $N\times N$ 的像素矩阵，

计算量 (计算距离的次数) 为 $(N-2)^2(4N)$，所以有一个 $O(N^3)$ 的计算复杂度。因此我们较常采用称为细化 (thinning) 的方法，在下列条件限制下，反复去除一物体的边缘点：① 不可去除端点；② 不可破坏连通性；③ 不可造成边缘过度腐蚀。

以下介绍一个针对二值化区域的细化方法。设 1 代表区域，0 代表背景。考虑一个边界点 p_1 及其 8 连通的点，如图 9.1-7 所示。首先判断至少有一个 8 连通为 0 的边界点是否要删除，而判定该删除的条件如下 (必须同时满足)：

$$\begin{aligned}&(1)\ 2 \leqslant N(p) \leqslant 6\\&(2)\ S(p)=1\\&(3)\ p_2 \cdot p_4 \cdot p_6 = 0\\&(4)\ p_4 \cdot p_6 \cdot p_8 = 0\end{aligned} \qquad (9.1\text{-}9)$$

p_9	p_2	p_3
p_8	p_1	p_4
p_7	p_6	p_5

图 9.1-7　p_1的邻近点的示意图

其中 $N(p)$ 为 p 的非零相邻点的个数，即 $N(p)=\sum_{i=2}^{9} p_i$，而 $S(p)$ 为 $p_2, p_3, \cdots, p_8, p_9$ 的顺序中 0 到 1 转变的次数。被判定为该删除的点暂不删除，但是要加以记录。等所有边界点都被判定完后，再一口气将所有该删除的点删除。接下来考虑第二阶段的删除步骤，其中 (1) 与 (2) 的删除条件与上一阶段相同，但 (3) 与 (4) 则改成

$$\begin{aligned}&(3)\ p_2 \cdot p_4 \cdot p_8 = 0\\&(4)\ p_2 \cdot p_6 \cdot p_8 = 0\end{aligned} \qquad (9.1\text{-}10)$$

同第一阶段，判定要删除的点只是加以记录而暂不删除，等最后同时删除。反复执行第一与第二阶段的程序，直到都没有任何可删除点为止，最后剩下没有被删除的点就是所要找的骨架。

条件 (1) 中的目的在于防止端点 $(N(p)=1)$ 被删除，或区域被过度腐蚀 $(N(p)=7)$。其他条件也都各有其原因，此问题留做习题。

9.2　边界描述子

9.2-1　形状数

在 9.1 节中曾提到链码是一个可用来表示边界的方法，但是它并不适合作为物体的描述子，因为它显然对于旋转的变化极为敏感，亦即同一物体旋转不同角度时，可能会得到两组截然不同的链码。尽管如此，若对链码进行某些归一化的处理，所得的码仍有可能成为描述子。形状数 (shape number) 即为其中一例。

给一个四方向的链码，方向数定义成差分链码中数值最小者。方向数的阶次 n 则是指其数字的个数而言，此外对一封闭的边界，n 必为偶数。两个相邻四方向链码数字 i 与 j 的差分链码数字 d 是指

$$d=(j-i+4)\bmod 4,\quad 0\leqslant i,j\leqslant 3 \tag{9.2-1}$$

图 9.2-1 显示 $n=6$ 的方向数的形成，其中在形成差分链码时是把链码视为头尾相接的环形序列。如果将图 9.2-1 的图形旋转 90 度，可发现其差分链码不会改变，显示差分链码对旋转不敏感。当然旋转对本身就是差分链码之一的方向数也不会有影响。

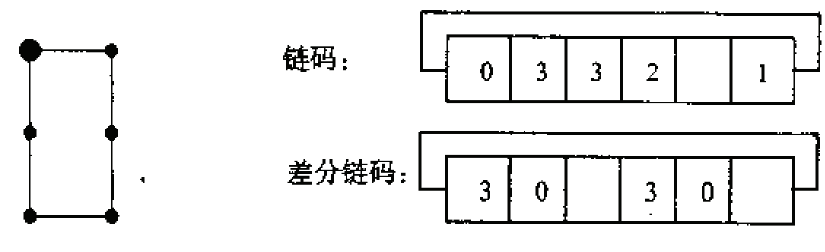


图 9.2-1　6 阶方向数的形成

以上的方向数配合以下讨论的格点方向归一化处理，获得对大小、平移和旋转均不敏感的实用描述子。考虑图 9.2-2 所示的物体边界。首先以两个相距最远的点的连线方向作为主轴，如图 9.2-2 中的虚线所示。找一矩形，其边对准主轴方向，其大小恰能把所有边界包括在内。将图 9.1-1 (a) 的四方形链码转一角度使其数字 “0” 所指的方向与主轴的方向一致。要获得 n 阶方向数，可考虑所有正整数 N_1 与 N_2 使得 $n/2=N_1+N_2$，且 N_2/N_1 最接近矩形长对宽的比例。例如图 9.2-2 中，$n=14$，此时 $(N_1,N_2)=(1,6)$、$(2,5)$ 或 $(3,4)$，其中 $(3,4)$ 最符合物体边界的矩形的长宽比例，故选 $N_1=3$，$N_2=4$。所对应的链码、差分链码及方向数亦显示于图 9.2-2 中。

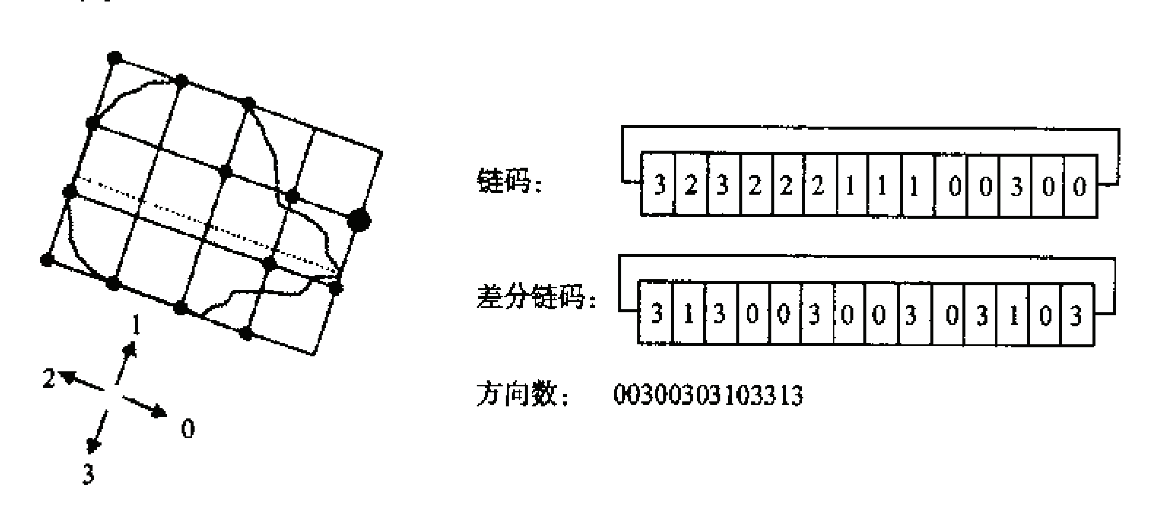


图 9.2-2　方向数产生的过程

9.2-2　傅立叶描述子

一个区域的周边为一封闭曲线。设以参数方程 $\{x(s),y(s)\}$ 来表示此曲线，其中参数 s 为相对于曲线上一个起始点 $[x(0),\ y(0)]$ 的沿线弧长。设 P 代表此区域的周长，则 $x(s)$ 与 $y(s)$ 各

可视为周期为 P 的周期函数，即 $x(s)=x(s+P)$、$y(s)=y(s+P),\forall s$。若进一步将图像平面看成复数平面且表示成 $z=x+\mathrm{i}y$，则区域周边变成一个复数的周期函数，即

$$z(s)=z(s+P) \tag{9.2-2}$$

对其以复数傅立叶级数展开可得

$$\begin{aligned} z(s) &= \sum_{k=-\infty}^{\infty} Z_k \mathrm{e}^{2\pi\mathrm{i}ks/P} \\ Z_k &= \frac{1}{P}\int_0^P z(s)\,\mathrm{e}^{2\pi\mathrm{i}ks/P}ds, \quad k=0,\pm1,\pm2,\cdots \end{aligned} \tag{9.2-3}$$

Z_k即为边界的傅立叶描述子 (Fourier descriptor、FD)。由于我们实际上面对的是离散的点 (设为$z_1,z_2,\cdots,z_N$)，因此将 (9.2-3) 式变成 N 点的离散傅立叶变换

$$\begin{aligned} z_r &= \sum_{k=0}^{N-1} Z_k \mathrm{e}^{\mathrm{i}2\pi rk/N} \\ Z_k &= \frac{1}{N}\sum_{k=0}^{N-1} z_r \mathrm{e}^{-\mathrm{i}2\pi rk/N} \end{aligned} \tag{9.2-4}$$

其中$r,k=0,1,\cdots,N-1$。N通常选为 2 的乘幂，如此 FFT 可用来加快运算速度。若舍去部分Z_k的高频系数，对其重建回来的点集合$\{\hat{z}_r\}$所形成的边界外形不会有太大的改变，因此选取少许的 Z_k就可用来区别外形不同的物体。

Z_k实际上会随物体平移、旋转等动作而改变，但是其改变是有规律且可预测的，例如物体旋转一个角度θ，则整个 Z_k都会乘上 $\mathrm{e}^{\mathrm{i}\theta}$，其他动作所造成的改变如表 9.2-1 所示。

表 9.2-1　傅立叶描述子的一些性质

动　作	边　界	傅立叶描述子
旋　转	$z_r\mathrm{e}^{\mathrm{i}\theta}$	$Z_k\mathrm{e}^{\mathrm{i}\theta}$
平　移	$z_r+(\Delta x+\mathrm{i}\Delta y)$	$Z_k+(\Delta x+\mathrm{i}\Delta y)\delta(k)$
大　小	αz_r	αZ_k
起始点改变	z_{r-m}	$Z_k\mathrm{e}^{-\mathrm{i}2\pi mk/N}$

9.3　区域描述子

9.3-1　一些简单描述子

有一些与区域有关的简单描述子如下：

厚度 (thickness)：让此区域消失所需最小的腐蚀次数。

最大弦 (maximum chord)：连结边界上相距最远的两点所形成的线段。

最小弦 (minimum chord)：与最大弦垂直的一个线段，其长度与最大弦的长度所形成的矩形恰能将整个区域围住。

直径 (diameter)：最大弦的长度。

延展度 (elongation)：最大弦与最小弦的长度比。

厚度与所采用的腐蚀方法有关。此参数大致上是平移与旋转不变的，但是随区域大小比例的改变而变。一个骨架型的区域与一个圆形区域，显然在厚度上会有明显的不同。延展度这个特征可用来区别较细长的物体与较方正或圆形的物体。

9.3-1 拓朴属性

拓朴的形状属性 (topological shape attributes) 是形状的一种特性，它是指一物体经由**橡皮板变换** (rubber-sheet transform) 后，其特性仍然不变。这种橡皮板变换可以想像成将一图像类似橡皮板一样拉长，图像内的物体也随着图像被拉长而产生空间上的形变。作此变换时不允许切割物体或任意连结物体。

在前述这种拓朴属性的描述下，距离观念很明显地不属于拓朴属性，因在拉长橡皮板的过程中距离早已产生变化。另外，水平和垂直的观念也不属于拓朴的性质，因在变换中并不考虑物体是否有旋转的效应。

接下来我们要讨论的是，在图像中物体的组成数 C 与物体中的洞数 H 间有一基本的关系，我们称之为**尤拉数** (Euler number)，其定义如下

$$E = C - H \tag{9.3-1}$$

尤拉数也是一种拓朴特性，因为 C 和 H 都有拓朴属性。

不规则的物体能以它们的拓朴特性来描述。首先我们引进**凸壳** (convex hull) 与**非凸状** (convex deficiency) 这两个名词。所谓凸壳是指拉长橡皮板中包围橡皮板的部分称为凸壳。非凸状又分为**湖状** (lakes) 与**堤防状** (bays) 两种；湖状是指完全被物体包围住的区域、堤防状是指物体与凸壳周边所包围的区域。我们将上述名词用图 9.3-1 所示的例子来说明。在有些应用中，以凸起的概念来描述物体比直接描述物体更简单。

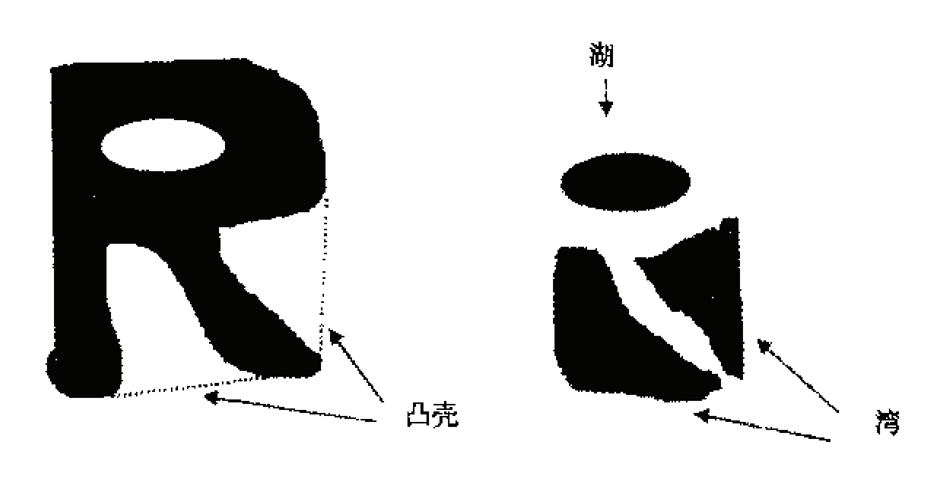


图 9.3-1 展示凸起概念的一个例子

9.3-1　距离、周长、面积的测量

假设有二点 (m_1,n_1) 与 (m_2,n_2)，二点间的距离有以下几种表示式：

(1) 欧几里德 (Euclidean) 距离

$$d_E=\left[(m_1-m_2)^2+(n_1-n_2)^2\right]^{\frac{1}{2}} \tag{9.3-2}$$

(2) 大小 (magnitude) 距离

$$d_M=|m_1-m_2|+|n_1-n_2| \tag{9.3-3}$$

(3) 最大值 (maximum value) 距离

$$d_X=\max\left\{|m_1-m_2|,|n_1-n_2|\right\} \tag{9.3-4}$$

接下来我们所探讨的周长与面积只针对二值图像才有意义。

一个物体的周长是指：在物体的边缘上由任一点开始围绕着边缘回到起始点的总像素个数。而一个物体包围的面积是指：物体边缘内的总像素和，其中包括所有 0 与 1 的点。

Gray[1971]曾提出一个有系统的方法来计算二值图像物体的面积与周长。他的做法是利用一组二值的样本图像来与所求的图像作比对，就可轻易地经由比对的结果来表示所求图像的面积与周长。

我们现在考虑 2×2 点像素的样本图像称为四位元组 (bit quads)，如图 9.3-2 所示。经由比对图像与四位元组的结果我们可分别求得其面积及周长如下：

(1) 面积

$$A_0=\frac{1}{4}\left[n\{Q_1\}+2n\{Q_2\}+3n\{Q_3\}+4n\{Q_4\}+2n\{Q_D\}\right] \tag{9.3-5}$$

(2) 周长

$$P_0=n\{Q_1\}+n\{Q_2\}+n\{Q_3\}+2n\{Q_D\} \tag{9.3-6}$$

其中 $n\{Q\}$ 代表图像与样本图像 Q 比对吻合的个数。

建立距离、周长、面积这些参数后，我们就可以开始对一物体的几何属性做探讨。首先，我们先假设在图像中物体的数目远大于洞的数目，也就是 $E \doteq C$。

在上面的前提下，我们可定义出似圆性 (circularity)

$$C_0=\frac{4\pi A_0}{(P_0)^2} \tag{9.3-7}$$

如果是一个圆形的物体，它的似圆性会刚好等于 1，其余物体的似圆性都将小于 1。

一幅图像内如果包含许多物体，但只有少许的洞，也就是如同前面的假设 $E \doteq C$，我们可推导出每个物体的平均周长及面积。

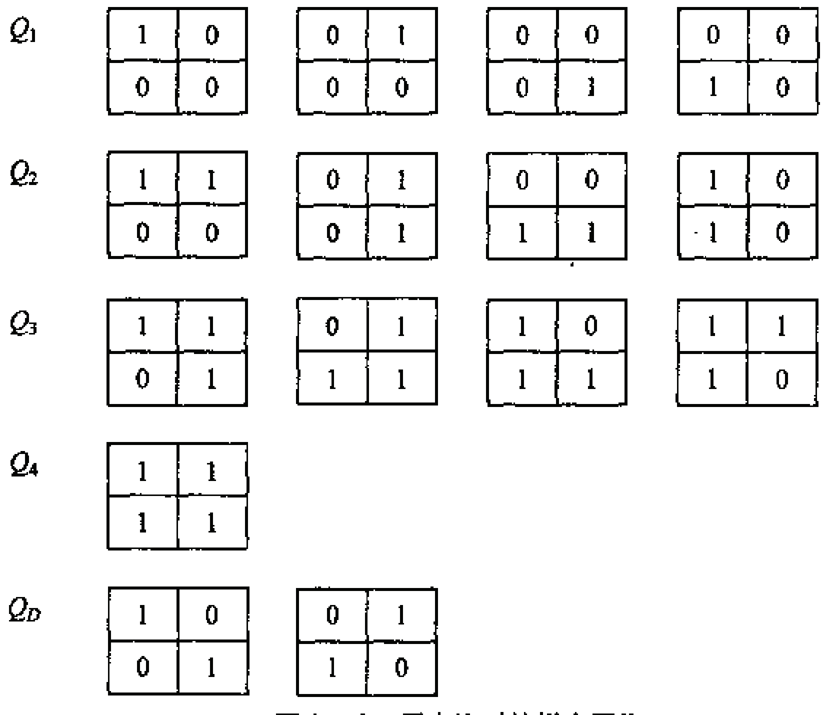


图 9.3-2 用来比对的样本图像

$$\text{平均面积：} A_A = \frac{A_0}{E} \tag{9.3-8}$$

$$\text{平均周长：} P_A = \frac{P_0}{E} \tag{9.3-9}$$

如果图像内包含瘦长的物体 (例如手写字)，我们也可求其近似的平均长度及宽度。

$$\text{平均长度：} L_A = \frac{P_A}{2} \tag{9.3-10}$$

$$\text{平均宽度：} W_A = \frac{2A_A}{P_A} \tag{9.3-11}$$

以上这些简单的测量对于识别图像的整体特征非常有用。

9.3-4 空间矩

在概率论中，对于连续函数 $f(x,y)$，其 $(k+l)$ 阶矩 (moment) 定义为

$$M_{k,l} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^k y^l f(x,y)\mathrm{d}x\mathrm{d}y \tag{9.3-12}$$

其中 $k,l = 0,1,2,\cdots$。而中心矩 (central moment) 则定义为

$$\mu_{k,l} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (x-\overline{x})^k (y-\overline{y})^l f(x,y)\mathrm{d}x\mathrm{d}y \tag{9.3-13}$$

其中 $\bar{x}$ 与 $\bar{y}$ 分别为 x 及 y 的期望值。

一个与平面区域有关的几何特性像是大小、位置、方向及形状等，其中很多特性与矩这个参数有关。在概率论中，矩用来呈现概率密度函数的特性，例如期望值是一阶矩、方差及协方差是二阶的中心矩等。在这里我们沿用相同的定义，但是将概率密度函数换成是二维二值图像的图形。一个二值图像 $b(m,n)$ 区域的 $(k+l)$ 阶矩定义成

$$M_{k,l}=\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}m^k n^l b(m,n) \tag{9.3-14}$$

把 $b(m,n)$ 换成灰度图像 $f(m,n)$ 的定义也有，但是其物理意义将不再是几何参数。例如一个概率密度函数 $f(m,n)$ 的零阶矩代表的是该函数所含盖的体积，而对 $b(m,n)$ 而言则变成我们所感兴趣的面积，也就是区域内所含像素的个数。

在定义过 $b(m,n)$ 的矩后，同理可定义其中心矩如下

$$\mu_{k,l}=\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}(m-\bar{x})^k(n-\bar{y})^l b(m,n) \tag{9.3-15}$$

其中

$$\bar{x}=\frac{M_{1,0}}{M_{0,0}},\quad \bar{y}=\frac{M_{0,1}}{M_{0,0}}$$

点 $(\bar{x},\bar{y})$ 称为*重心* (center of gravity) 或*质心* (centroid)。若一般矩为已知，则中心矩的计算可通过一般矩来达成，例如

$$\begin{aligned}
\mu_{0,0}&=M_{0,0}\\
\mu_{0,1}&=\mu_{1,0}=0\\
\mu_{2,0}&=M_{2,0}-\bar{x}M_{1,0}\\
\mu_{1,1}&=M_{1,1}-\bar{y}M_{1,0}\\
&\vdots
\end{aligned} \tag{9.3-16}$$

二阶的中心矩的许多特性相当于概率论中的协方差矩阵，以及力学中物体旋转的转动惯量。此协方差矩阵为

$$\boldsymbol{U}=\begin{bmatrix}\mu_{2,0} & \mu_{1,1}\\ \mu_{1,1} & \mu_{0,2}\end{bmatrix} \tag{9.3-17}$$

对角化后，可求出对角矩阵

$$\boldsymbol{E}^{\mathrm{T}}\boldsymbol{U}\boldsymbol{E}=\boldsymbol{\Lambda} \tag{9.3-18}$$

其中

$$\boldsymbol{E}=\begin{bmatrix}e_{1,1} & e_{1,2}\\ e_{2,1} & e_{2,2}\end{bmatrix} \tag{9.3-19}$$

且矩阵 $\boldsymbol{E}$ 的行向量为 $\boldsymbol{U}$ 的特征值向量，另外

$$\boldsymbol{\Lambda}=\begin{bmatrix}\lambda_1 & 0\\ 0 & \lambda_2\end{bmatrix} \tag{9.3-20}$$

含 $\boldsymbol{U}$ 的特征值。注意到 (9.3-18) 式与 (4.8-10) 式有相同的含义。从 (9.3-17) 式可得其特征值为

$$\begin{aligned}\lambda_{\max}&=\frac{1}{2}(\mu_{2,0}+\mu_{0,2})+\frac{1}{2}\sqrt{\mu_{2,0}^2+\mu_{0,2}^2-\mu_{0,2}\mu_{2,0}+4\mu_{1,1}^2}\\ \lambda_{\min}&=\frac{1}{2}(\mu_{2,0}+\mu_{0,2})-\frac{1}{2}\sqrt{\mu_{2,0}^2+\mu_{0,2}^2-\mu_{0,2}\mu_{2,0}+4\mu_{1,1}^2}\end{aligned} \tag{9.3-21}$$

则区域的方向角可表示成

$$\theta=\tan^{-1}\left\{\frac{\lambda_{\max}-\mu_{2,0}}{\mu_{1,1}}\right\} \tag{9.3-22}$$

此外、一个区域的离心率 (eccentricity) 可定义成

$$\sqrt{\frac{\lambda_{\max}}{\lambda_{\min}}} \tag{9.3-23}$$

此参数与比例大小及方向无关、只与形状有关。

由于 $\mu_{0,0}=M_{0,0}$ 为区域的面积，故可作为区域大小的量测。并可以此来求得一个归一化的矩，而获得一个与比例大小无关的描述。由 (9.3-13) 式可推得，对一除上比例因子 $\sqrt{\alpha}$ 的函数 $f\left(x/\sqrt{\alpha},y/\sqrt{\alpha}\right)$，其中心矩为 $f(x,y)$ 的中心矩的 $\left(\sqrt{\alpha}\right)^{k+l+2}$ 倍。因为 $\mu_{0,0}$ 为面积，故可将 $\sqrt{\mu_{0,0}}$ 视为此比例因子、因此可得归一化的中心矩为

$$\eta_{k,l}=\frac{\mu_{k,l}}{\left(\sqrt{\mu_{0,0}}\right)^{k+l+2}} \tag{9.3-24}$$

由归一化的第二与第三阶矩可导出下面七个不变矩 (moment invariant)：

$$\phi_1=\eta_{2,0}+\eta_{0,2} \tag{9.3-25}$$

$$\phi_2=(\eta_{2,0}-\eta_{0,2})^2+4\eta_{1,1}^2 \tag{9.3-26}$$

$$\phi_3=(\eta_{3,0}-3\eta_{1,2})^2+(3\eta_{2,1}-\eta_{0,3})^2 \tag{9.3-27}$$

$$\phi_4=(\eta_{3,0}+\eta_{1,2})^2+(\eta_{2,1}+\eta_{0,3})^2 \tag{9.3-28}$$

$$\begin{aligned}\phi_5=&(\eta_{3,0}-3\eta_{1,2})(\eta_{3,0}+\eta_{1,2})\left[(\eta_{3,0}+\eta_{1,2})^2-3(\eta_{2,1}+\eta_{0,3})^2\right]\\ &+(3\eta_{2,1}-\eta_{0,3})(\eta_{2,1}+\eta_{0,3})\left[3(\eta_{3,0}+\eta_{1,2})^2-(\eta_{2,1}+\eta_{0,3})^2\right]\end{aligned} \tag{9.3-29}$$

$$\phi_6 = (\eta_{2,0} - \eta_{0,2})\left[(\eta_{3,0} + \eta_{1,2})^2 - (\eta_{2,1} + \eta_{0,3})^2\right] + 4\eta_{1,1}(\eta_{3,0} + \eta_{1,2})(\eta_{2,1} + \eta_{0,3}) \tag{9.3-30}$$

$$\phi_7 = (3\eta_{2,1} - \eta_{0,3})(\eta_{3,0} + \eta_{1,2})\left[(\eta_{3,0} + \eta_{1,2})^2 - 3(\eta_{2,1} + \eta_{0,3})^2\right] + (3\eta_{1,2} - \eta_{3,0})(\eta_{2,1} + \eta_{0,3})\left[3(\eta_{3,0} + \eta_{1,2})^2 - (\eta_{2,1} + \eta_{0,3})^2\right] \tag{9.3-31}$$

此组不变矩不受平移、旋转及大小比例改变的影响 (Hu[1962])。稍后有人注意到这些不变矩的动态范围有时会很大，因此建议采用 $\log|\phi_i|$ (Hsia[1981])。若 ϕ_i 的正负号很重要，则用 $\mathrm{sgn}|\phi_i|\log|\phi_i|$。

不变矩对二维物体识别非常有用，曾有人用在飞机识别上 (Dudani 等人[1977])，也曾被推广到三维物体的识别上 (Reeves 与 Taylor[1989])，还被用来决定医学图像中三维结构体的方向 (Faber 与 Stoklly[1988])。不过要注意的是这些不变矩并不足以区别所有形状，而且对噪声很敏感。

9.3-5　纹理组织

纹理组织 (texture) 是另一种描述区域的重要方法。一般天然纹理组织较具随机性，而人造的则较有规律或是有周期性。通常我们采用像细致与粗糙程度、对比程度、方向特性及规则性来描述纹理组织。所有描述的方法大致可分成三种：统计法、结构法及变换法 (或频谱法)。具有随机特性的纹理组织很适合用统计法，例如将其以随机场 (random field) 描述。结构法在于描述基本图像原始结构 (image primitive) 的排列组合，例如以排列规则的平行线来描述一纹理组织。变换法或称频谱法是利用傅立叶变换的性质，试图找到频谱中能量集中的窄频谱以检测出纹理组织的周期性。

- **统计法**

(1) 自相关函数 (autocorrelation function，ACF)

从某个角度来看，纹理组织与基本图像单元的大小有关，亦即比较大的基本图像单元相当于比较粗糙的纹理组织，反之则代表比较细密的纹理组织。

$f(m,n)$ 的自相关函数定义成

$$r(m,n) = \sum_k \sum_l f(k,l) f(m+k, n+l) \tag{9.3-32}$$

其中 $|k| < L_k$ 且 $|l| < L_l$，而 L_k 及 L_l 为正值常数。若基本图像单元比较大，其 ACF 随着距离增加 (即 $|k|$ 与 $|l|$ 变大) 而缓慢衰减，反之则衰减较快。如果所碰到的纹理组织有空间上的周期性，则 ACF 也会起起落落呈现周期性。

ACF 缓慢衰减相当于其“扩散度”或“宽度”较宽，因此一个描述纹理组织粗糙程度的方式是量测 ACF 的“宽度”或“扩散度”，其中一个测量法是由 Faugeras 与 Pratt[1980]所提出的矩生成函数

$$M_{k,l}=\sum_{m}\sum_{n}(m-\eta_m)^k(n-\eta_n)^l r(m,n) \tag{9.3-33}$$

其中

$$\eta_m=\sum_{m}\sum_{n}mr(m,n),\quad \eta_n=\sum_{m}\sum_{n}nr(m,n) \tag{9.3-34}$$

在所有 $M_{k,l}$ 中通常用 $M_{2,0}$、$M_{0,2}$、$M_{1,1}$ 及 $M_{2,2}$。

(2) 边缘密度

一个纹理组织的粗糙程度也可以用边缘像素的密度来表示。所谓边缘密度是指在单位面积内边缘像素的平均个数，可表成

$$D(m,n)=\frac{1}{(2J+1)(2K+1)}\sum_{j=-J}^{J}\sum_{k=-K}^{K}E(m+j,n+k) \tag{9.3-35}$$

其中 $2J+1$ 与 $2K+1$ 为观察视窗的大小，而

$$E(j,k)=\begin{cases}1, & \text{若为边缘像素}\\ 0, & \text{不为边缘像素}\end{cases} \tag{9.3-36}$$

(3) 二阶中心矩

二阶中心矩 (亦即方差 σ^2) 对于纹理组织的描述格外重要，例如一个纹理组织的相对平滑性可表示成

$$R=1-\frac{1}{1+\sigma^2} \tag{9.3-37}$$

$R=0$ 代表 $\sigma^2=0$，亦即灰度强度完全一致的区域。若 σ^2 很大，则 R 趋近于 1。

(4) 灰度共同出现

一般直方图是对单一像素的个别灰度值统计，此处则是考虑一对像素的灰度值所形成的直方图，因此得到的是一个二维的直方图，可写成 $h(i,j)$，其中 i 与 j 代表有某个相关位置的两个像素的个别灰度值，$h(i,j)$ 则代表具有此种关系的像素对的个数。将直方图 $h(i,j)$ 除上总数得到 $p_{i,j}$，它可视为 (i,j) 对的联合概率密度的一个估测。由 $p_{i,j}$ 组成的矩阵 $\boldsymbol{P}=\{p_{i,j}\}$，称为灰度共同出现 (co-occurrence) 或相依 (dependency) 矩阵。

我们举一例来说明 $\boldsymbol{P}$ 矩阵的形成。设一图像的灰度只有 0,1,2 三种。考虑一个 5×5 的图像如下

$$\begin{matrix}1 & 2 & 1 & 0 & 0\\ 2 & 1 & 1 & 1 & 1\\ 0 & 0 & 0 & 1 & 0\\ 0 & 0 & 2 & 0 & 1\\ 2 & 2 & 1 & 2 & 2\end{matrix}$$

假设所考虑的位置关系是第二个像素在第一个像素右下方且相邻，则 $h(i,j)$ 所形成的矩阵可写成

$$\begin{matrix} 2 & 2 & 3 \\ 2 & 4 & 0 \\ 1 & 1 & 1 \end{matrix}$$

总数共计 16 个，故矩阵 $\boldsymbol{P}$ 为

$$\boldsymbol{P}=\begin{bmatrix} 1/8 & 1/8 & 3/16 \\ 1/8 & 1/4 & 0 \\ 1/16 & 1/16 & 1/16 \end{bmatrix}$$

在上述的例子中所选择的位置关系对 $-45°$ (或 $135°$) 有相同灰度的纹理组织会特别敏感 ($\max\limits_{i,j}(p_{i,j})=1/4$可视为一敏感度的指标)，选取不同的位置关系可用来检测具有不同特性的纹理组织。

从矩阵 $\boldsymbol{P}$ 中可获得以下几个纹理组织的特征：

$$\text{最大概率：}\max_{i,j}(p_{i,j}) \tag{9.3-38a}$$

$$k\text{ 阶差分矩：}\sum_i\sum_j (i-j)^k p_{i,j} \tag{9.3-38b}$$

$$\text{能量的均匀性：}\sum_i\sum_j p_{i,j}^2 \tag{9.3-38c}$$

$$\text{熵：}-\sum_i\sum_j p_{i,j}\log p_{i,j} \tag{9.3-38d}$$

- **结构法**

在纹理组织的结构模型中，纹理组织被看成是由较小的基本原始结构依某种规则排列组合而成。因此要描述一个纹理组织必须描述此原始结构及其置放的法则。原始结构可依灰度、形状及均匀性来设计，置放法则可采用周期性、邻接性及最近距离等观念来设计。

Carlucci[1972]建议用线段与多边形当原始结构，而置放方法则用像图形一般的语言来描述。Zucker[1976a，1976b]则把真实的纹理组织想成是某个理想的纹理组织的失真变形。Lu 与 Fu[1978]则采用树状语言文法结构。其他方法可见 Tomita 等人[1982]以及 Leu 与 Wee[1985]。

- **频谱法**

像粗糙、细致、方向等纹理组织的特征均可用傅立叶频谱来估测。由于纹理组织的粗糙程度与其空间上的周期 (spatial period) 成正比，故对有较粗糙的纹理组织的区域，其傅立叶频谱能量集中在低频上；反之较细致的区域的频谱能量会集中在较高频的地方。另外由最主要的频谱峰值的位置可看出纹理组织图样的主要方向。

9.4 形态学

形态学 (morphology) 的重点是研究一物体的形状或结构，也可说是探讨一物体内部之间的相互关系。在语言学的领域里，形态学是指对字词结构的一种研究；而在生物学中形态学则与生物或有机体的形状有关。例如由树叶的形状可用来辨认植物，又如由细菌的菌丛可分辨其种类等。

在数字图像处理的应用领域中，我们亦沿用形态学这个名词，但为了与其他应用领域有所区别，故有人称为数字形态学 (digital morphology) 或数学形态学 (mathematical morphology)。称数字形态学是因为处理数字图像的原故，称为数学形态学则是因为必须用到数学的概念，特别是有关集合论 (set theory) 方面的数学。

本节主要是介绍数学形态学中的几个概念。虽然有些概念，如一个区域的骨架及边界等在先前的章节中已提到过，但是此处我们将在形态学这个统一的理论结构下，对其重新再探讨。我们将探讨二值图像的形态学。

图 9.4-1 中所取的图像设为 B，将 B 的 3×3 矩阵与处理的结构元素 S 作逻辑运算得一输出点。取一新的 3×3 矩阵重复上述步骤。如此一直取完输入图像，这样就完成了处理过程。整个过程像是求二维的卷积。不同的结构元素与逻辑运算对图像有不同的效应。我们先做一些术语上的定义，再从最基本的腐蚀 (erosion) 以及膨胀 (dilation) 这两种运算讨论起。

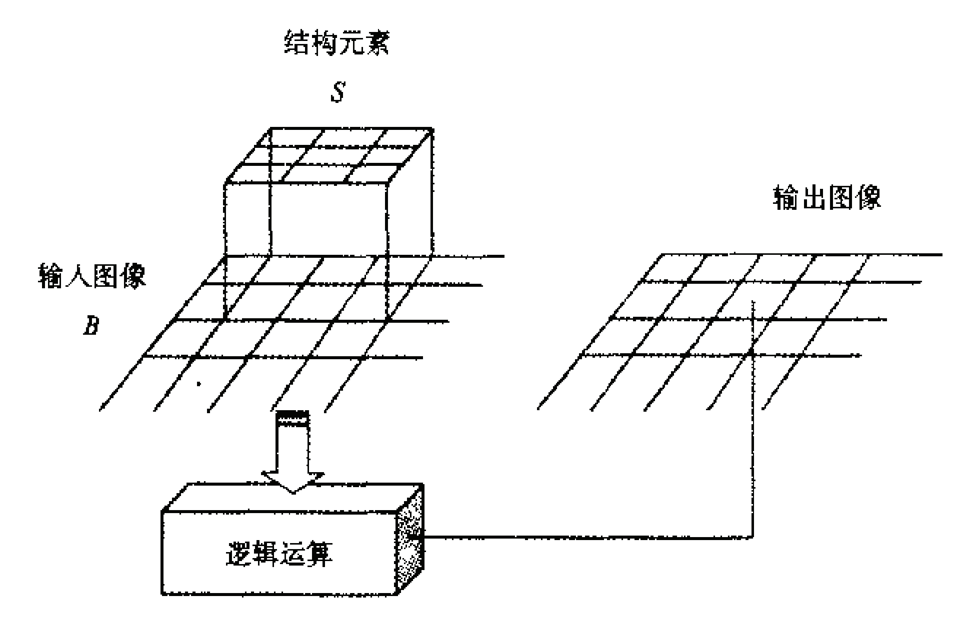


图 9.4-1 形态学操作想像图

设集合 B 与 S 分别含元素 $b=(b_1,b_2)$ 与 $s=(s_1,s_2)$，其中 b_i 与 s_i，$i=1,2$ 分别代表整数的坐标轴位置。定义 B 平移 (translation) $x=(x_1,x_2)$ 单位 (表示成 $(B)_x$)为

$$(B)_x=\{a\mid a=b+x,\ 对于 b\in B\} \tag{9.4-1}$$

定义 S 的反射或映像 (reflection) (表示成 $\hat{S}$)为

$$\hat{S}=\{x \mid x=-s,\ \text{对于} s\in S\} \tag{9.4-2}$$

定义集合 B 的补集 (complement) 为

$$B^c=\{x \mid x\notin B\} \tag{9.4-3}$$

最后定义集合 B 与 S 的差 (difference) 为

$$B-S=\{x \mid x\in B,\ x\notin S\} \tag{9.4-4}$$

- **腐蚀** (Erosion)

这是一种从物体的边界上，将物体往内收缩若干像素的方法，其数学表示为

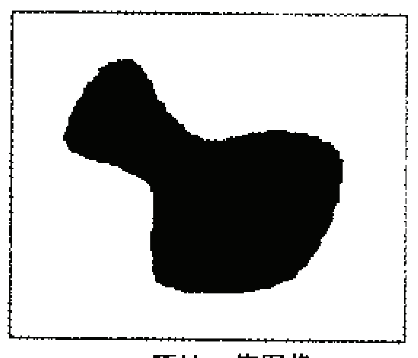
$$E=B\ominus S=\{x \mid (S)_x\subseteq B\} \tag{9.4-5}$$

- **膨胀** (Dilation)

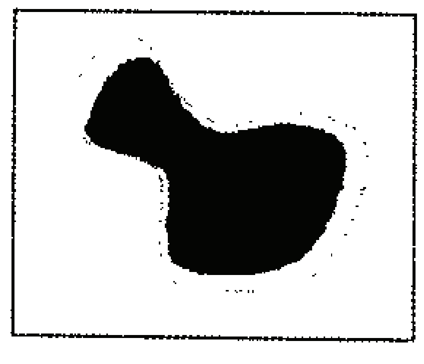
与腐蚀相反的操作，将物体的边界往外膨胀若干像素的方法，其数学表示为

$$D=B\oplus S=\{x \mid (\hat{S})_x\cap B\neq\varnothing \tag{9.4-6}$$

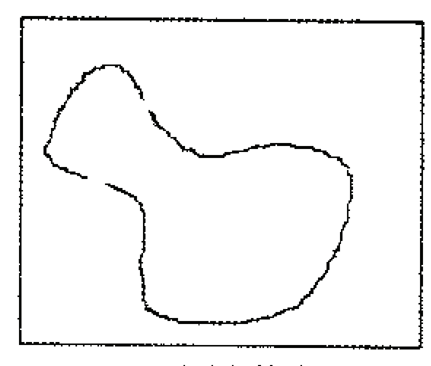
图 9.4-2 是表示腐蚀及膨胀对一幅图像所可能造成的结果，其中虚线代表原图像的边界，可看出经腐蚀过程后原物体内缩，而膨胀是向外扩张。这两个基本步骤很重要，因为往后的处理都是联合使用或交互使用这两个步骤。



(a) 原始二值图像



(b) 腐蚀的结果



(c) 膨胀的结果

图 9.4-2　腐蚀及膨胀对图像的影响

● **断开** (Opening)

定义为

$$B \circ S = (B \ominus S) \oplus S \tag{9.4-7}$$

意思是说先腐蚀再膨胀，或是重复腐蚀直到消除掉所有不想要的点或线，再用膨胀恢复原图形。这样的步骤可用来消除物体边界外的小分枝干扰，或者用来修剪物体多余的小分枝。

● **闭合** (Closing)

定义为

$$B \bullet S = (B \oplus S) \ominus S \tag{9.4-8}$$

这与断开是相反的操作，即先膨胀再腐蚀。与上面相同也可重复使用，但要注意在恢复图像时，膨胀过几次，收缩就需几次。在过程中，我们可用来将在物体内的小洞补回，或是修补二端点之间的缺口、细长的缺口、连结断线等。

图 9.4-3 显示一个断开与闭合的例子，其中所用的结构元素 S 为一小圆盘。从图 9.4-3 的 (a) 到 (e) 可发现，其实 (e) 的边界可由小圆盘 S 贴着 B 边界内滚动的 S 的边界所形成；同理 (i) 的边界可由 S 贴着 B 边界外滚动的 S 的边界所形成。

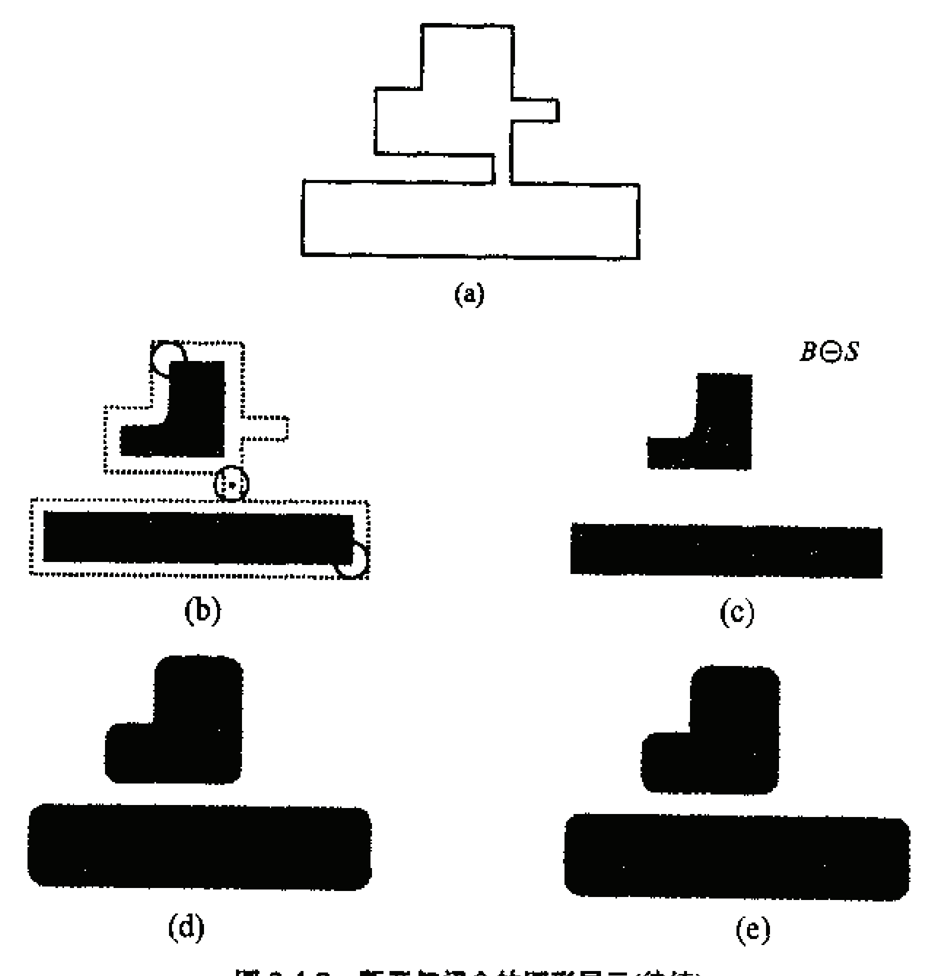


图 9.4-3　断开与闭合的图形展示(待续)

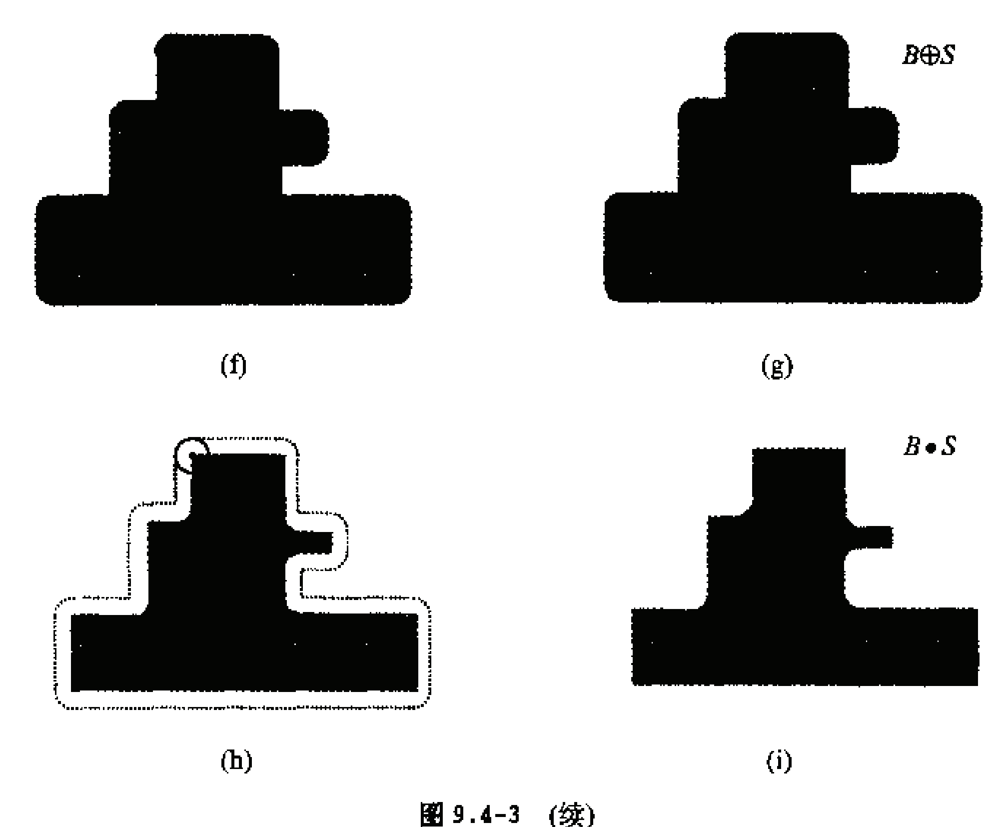


图 9.4-3 (续)

● **收缩** (Shrinking)

重复使用腐蚀的技巧，直到线宽都仅剩单像素的大小时为止。常用在计算整幅图像中物体的总数，例如有人把它用在石棉纤维的计数上。

● **细化**(Thinning)

也是重复使用腐蚀的技巧；但确保物体的连通性不被损坏。通常采用两个步骤：① 事先标示要移除的像素；② 在不破坏连通性的情况下将①的候选像素消除。细化可以将所感兴趣的物体缩小到只有单像素宽，如此呈现出物体的形状。

● **骨化**(Skeletonization)

骨化非常类似细化，但其结果并不相同，骨化可由 9.1-4 节中轴变换来定义。这种作法有以下几点好处：

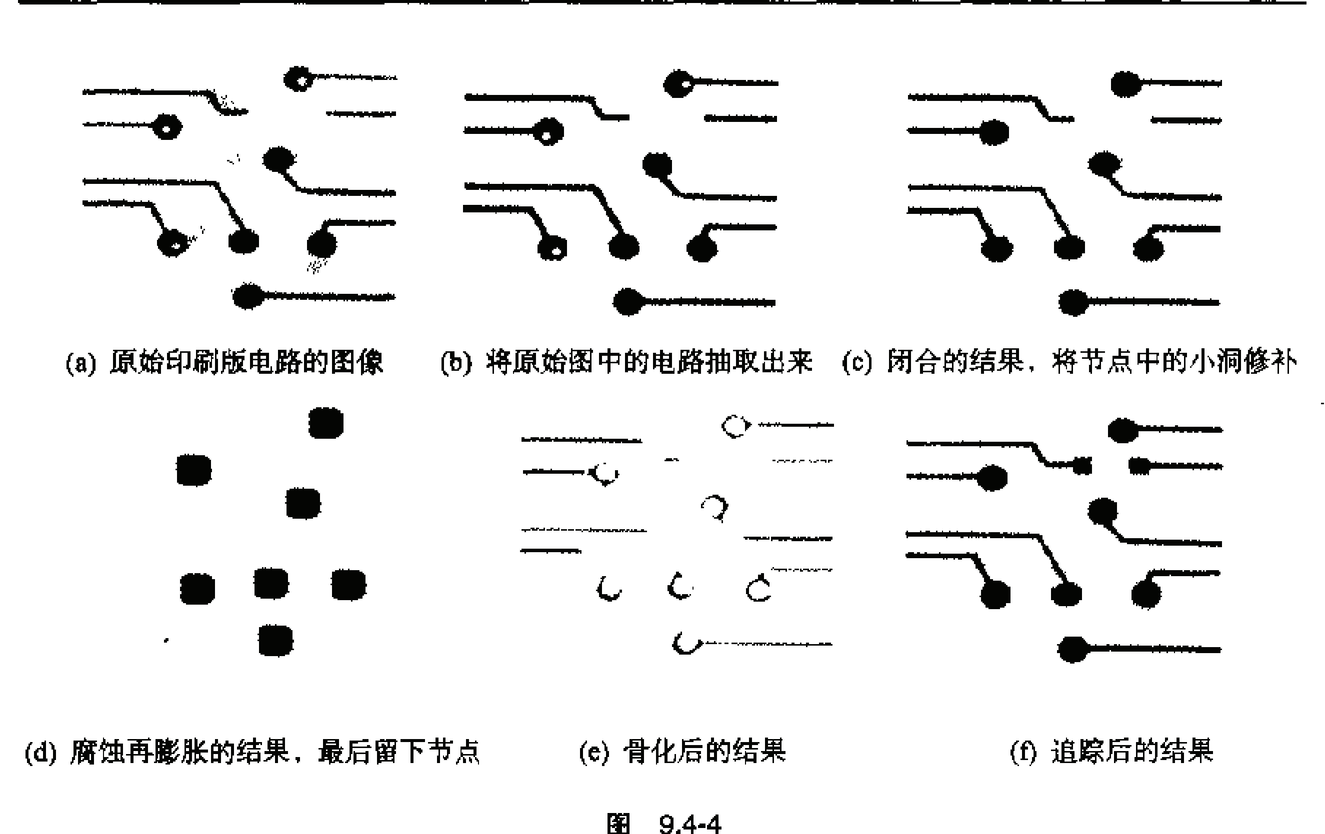
① 不会删除端点；

② 删除的点不会中断连接；

③ 不会过度腐蚀。

● **厚化**(Thickening)

这是和细化相反的操作，由重复使用膨胀所得的结果。与细化相同的是以两个步骤完成；要注意的是不可与周围的物体相合并。图 9.4-4 是一个形态学的应用实例，其目的是通过追踪后找出断点。



(a) 原始印刷版电路的图像　(b) 将原始图中的电路抽取出来　(c) 闭合的结果，将节点中的小洞修补

(d) 腐蚀再膨胀的结果，最后留下节点　(e) 骨化后的结果　(f) 追踪后的结果

图 9.4-4

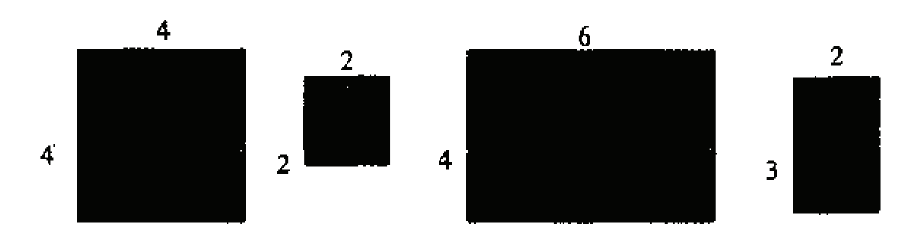
习　题

1. 为何要选定对大小、平移及旋转等变化均不敏感的描述子?
2. 找出所有阶次为 8 的形状及其方向数。
3. 对一个圆取直线近似的结果会是什么?
4. 证明三次的 B-样条函数有连续的一阶与二阶导数。
5. (1) 写一程序根据 MAT 定义找出一区域的中轴或骨架。
   (2) 以实际程序执行的时间估测其计算复杂度。
6. (9.1-9) 与 (9.1-10) 式中所列的各个条件有何意义?
7. (1) 以 9.1 节所提供的方法写一细化程序。
   (2) 以实际程序执行的时间估测其计算复杂度。
   (3) 此细化的结果一般是否会与 MAT 定义出的骨架相符?为什么?
8. 证明表 9.2-1 所示的傅立叶描述子的性质。
9. 下面这两种区域，你可以用何种简单的区域描述子加以区别?



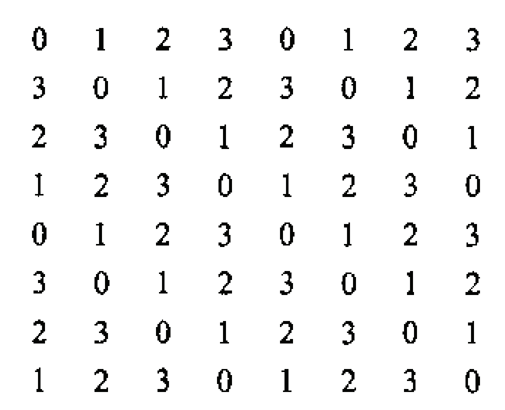
10. 分别找出下面两个字的凸壳及湖状与堤防状的非凸部分。

11. 举二个二值图像为例分别进行 9.3-3 节所示的各种量测结果。
12. 对以下四物体以程序或手算出其前二个不变矩 (即 ϕ_1及ϕ_2)：



其中的数字代表像素个数，比较并讨论其结果。

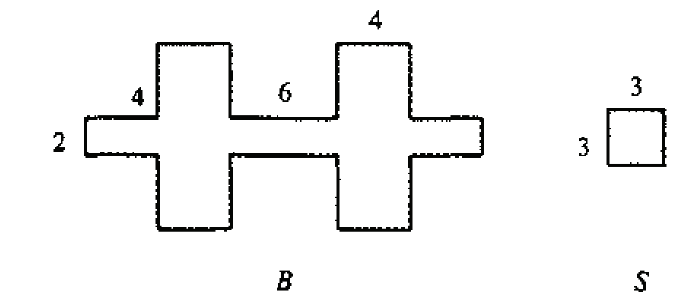
13. 设一图像有 0,1,2,3 四种灰度值。考虑一个 8×8 的图像如下：

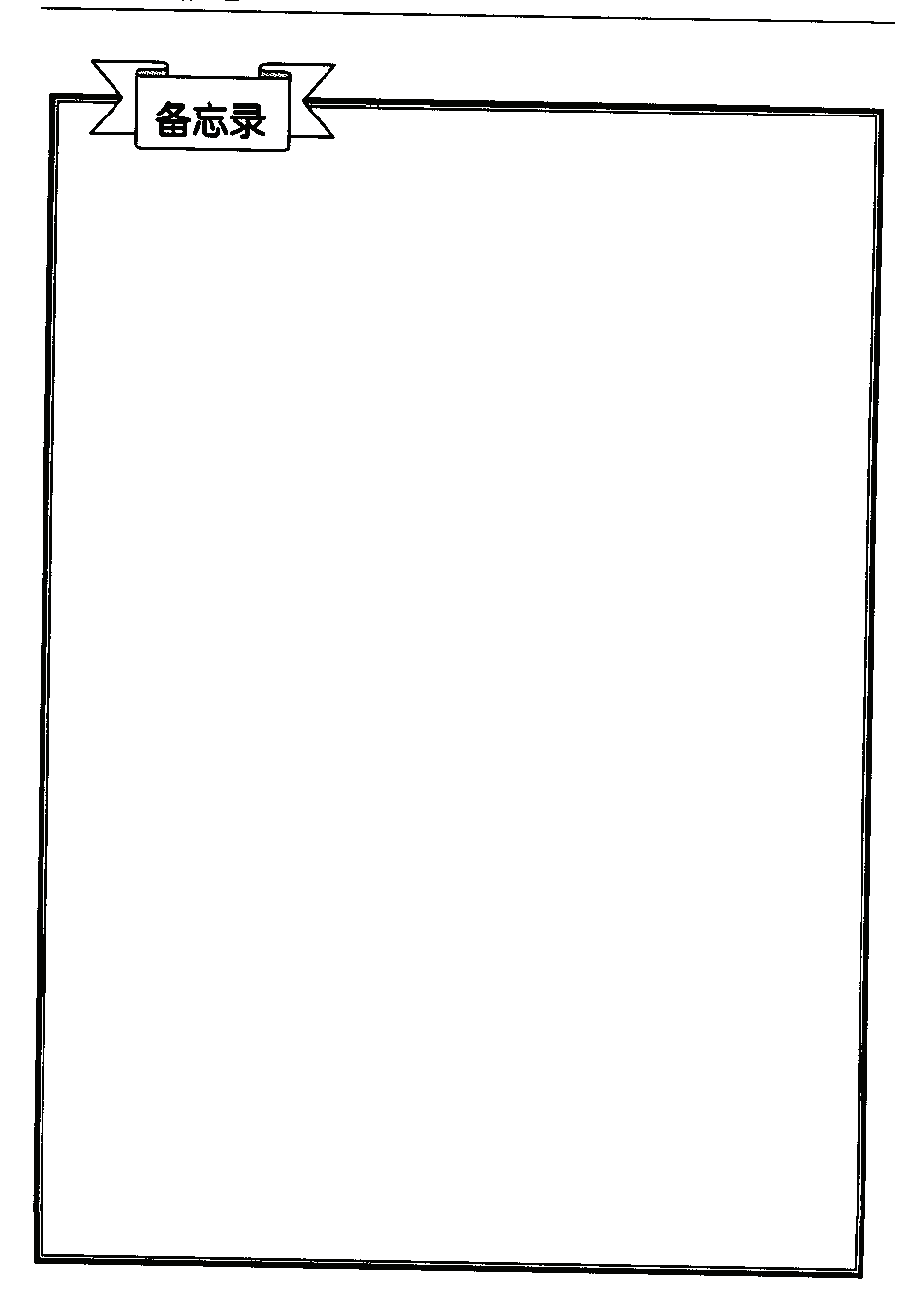


$$
\begin{matrix}
0 & 1 & 2 & 3 & 0 & 1 & 2 & 3 \\
3 & 0 & 1 & 2 & 3 & 0 & 1 & 2 \\
2 & 3 & 0 & 1 & 2 & 3 & 0 & 1 \\
1 & 2 & 3 & 0 & 1 & 2 & 3 & 0 \\
0 & 1 & 2 & 3 & 0 & 1 & 2 & 3 \\
3 & 0 & 1 & 2 & 3 & 0 & 1 & 2 \\
2 & 3 & 0 & 1 & 2 & 3 & 0 & 1 \\
1 & 2 & 3 & 0 & 1 & 2 & 3 & 0
\end{matrix}
$$

设位置关系为“第一个像素在第二个像素的左上方且相邻”，试求其共同出现矩阵及其最大概率。

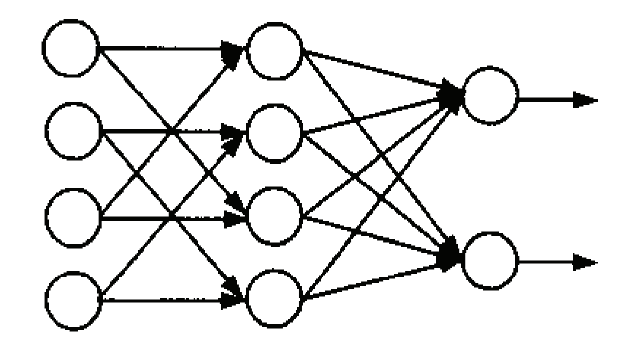
14. 对下列的图像 B 与结构元素 S 执行各运算：
    (1) 腐蚀
    (2) 膨胀
    (3) 断开
    (4) 闭合




备忘录

第十章

图像模式识别



本书的最后一部分为图像模式识别，属于整个数字图像处理的高层处理部分(如第八章图 8.0-1 所示)。这个过程很类似于一般用智慧认知这一术语所指的过程。一般实际的图像模式识别系统，都针对其应用对象有特别的“领域知识”或解决方案，而很难有一个可解决众多不同类问题的通用图像模式识别系统。然而，有一些基本原理或原则，对多数的图像模式识别系统均适用，本章的主要目的就是讨论这些基本原理。

10.1 分 类

● 图像模式与图像模式类别

图像模式 (pattern) 可以说是我们日常生活当中获取信息的主要形式，只要睁开眼睛，放眼所及的东西都可视为图像模式，例如每天读书看报所接触的文字就是一个很标准的图像模式。另外五线谱上的音符、化学结构的表示式以及心电图等也都是。

同样，只要我们睁开眼睛，就会不自觉地执行图像模式识别，例如我们可轻易辨认出家人及亲朋好友并叫出其姓名。我们是如何做到的?用电脑如何来模拟我们这种天生的能力?显然必须先具备抽取图像模式的特征 (feature) 的能力，再根据这些特征来区分不同的图像模式类型。

图像模式可以是一组量测或观察的结果，它可以用定量或定性的方式来描述，甚至也可用向量或矩阵来代表。一般而言，图像模式是由若干个像第九章中所述的描述子所形成。例如，不变矩的描述子可写成下列向量

$$\Phi=[\phi_1 \quad \phi_2 \quad \phi_3 \quad \phi_4 \quad \phi_5 \quad \phi_6 \quad \phi_7]^{\mathrm{T}} \tag{10.1-1}$$

上述向量称为特征向量 (feature vector)。对不同物体形状的图像模式，其特征向量通常会有显著的不同，但对同一形状物体的旋转、放大及缩小版的特征向量而言却大致相近，这就是图像模式分类的一个依据。

● 分类器的设计

分类器的设计理念与图像模式类别的特性息息相关。当图像模式类别是依照点名查表的方式对号入座时，则此分类器的设计就变成单纯的模版比对 (template matching)。例如有固定大小的 26 个英文字母为模版，输入一个未知的图像模式直接比对就可知其为哪一个英文字母，当然先决条件是输入的字几乎不能有任何的失真，否则比对会有困难。当图像模式是依照有某种共同的特性组成一类时，分类器设计的重点变成是检测及处理这些共同的特征。一般而言，对于一个图像模式差异程度大的图像模式识别问题，这个设计理念比简单的模版比对好，因为储存特征通常比直接储存比对图像模式有较小的储存需求，而且处理起来比较快速，也有较高的失真容忍度。

一般来说，分类器将对每一输入向量计算出一个对应的数值，而此数值将会指出此输入图像模式属于哪一个类别。大多数的分类器决策规则都被简化成一个临界值规则，这个临界

值将测试空间分割为数个不连续的空间，其中每个空间区域分别代表不同的单一类别。如果特征向量落在某一特定区域上，则此物体将会被归类成某一特定对应类别。

在建立分类器基本决策规则之后，就必须找出区分各类别的特定临界值。一般做法是使用一群已知其类别归属的特征样本来训练分类器，这样的一群特征样本称为训练集 (training set)。训练集是由一些预先精确分类，选自不同类别具有各类别代表性的样本所组成。训练集逐步调整决策平面 (decision surface)，使分类器在训练集中作用时正确率为最大。

考虑图 10.1-1 所示的一个分类状况，此处共有两类 (即“∇”与“◊”)，每个图像模式的特征都被表示成一个二维的特征向量 $\boldsymbol{x}=[x_1 \quad x_2]^{\mathrm{T}}$，被画成一个“∇”或“◊”。图中黑线的方程式是 $x_1=x_2$，因此其分类工作可转变成使用一个决策函数 $D(\boldsymbol{x})=x_2-x_1$，而其决策法则是以零为临界值，即若 $D(\boldsymbol{x})>0$，则此 $\boldsymbol{x}$ 属于“∇”这一类，若 $D(\boldsymbol{x})<0$ 则此 $\boldsymbol{x}$ 属于“◊”这一类。

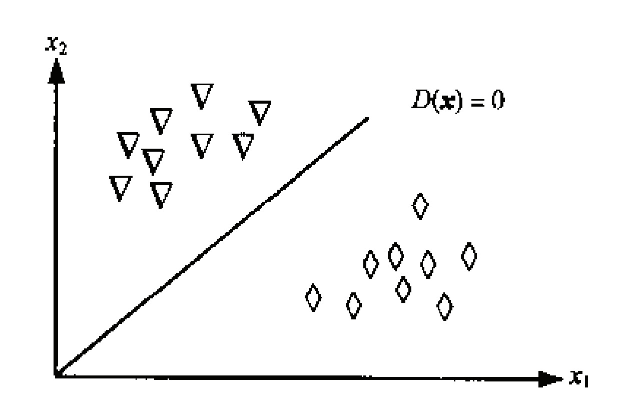


图 10.1-1　一个简单的分类器设计

分类器所用的决策函数 $D(\boldsymbol{x})$ 该如何通过训练集来决定呢？这里介绍一个简单的迭代法。假设有两个训练集分属于二个不同的类别 ω_1 与 ω_2。设 $D_k(\boldsymbol{x})$ 代表第 k 次迭代时所获得的决策函数，且

$$D_k(\boldsymbol{x})=\boldsymbol{w}_k^{\mathrm{T}}\boldsymbol{x}_k \tag{10.1-2}$$

其中 $\boldsymbol{w}_k$ 为第 k 次迭代时的权重向量，$\boldsymbol{x}_k$ 为第 k 个输入向量。$\boldsymbol{w}_1$ 是任选的初始权重向量。在第 k 次时的训练法则如下

$$\begin{cases} \boldsymbol{w}_{k+1}=\boldsymbol{w}_k+c\,\boldsymbol{x}_k, & 若\boldsymbol{w}_k^{\mathrm{T}}\boldsymbol{x}_k\leqslant 0\text{且}\boldsymbol{x}_k\in\omega_1 \\ \boldsymbol{w}_{k+1}=\boldsymbol{w}_k-c\,\boldsymbol{x}_k, & 若\boldsymbol{w}_k^{\mathrm{T}}\boldsymbol{x}_k\geqslant 0\text{且}\boldsymbol{x}_k\in\omega_2 \\ \boldsymbol{w}_{k+1}=\boldsymbol{w}_k, & 其他情况 \end{cases} \tag{10.1-3}$$

其中 c 为一正值修正增量。简言之，若分类错误则改变权重向量，使决策平面改变，否则决策平面维持不动。此训练法对于像图 10.1-1 这种所谓线性可分离的情况保证收敛，换言之在有限次的训练次数下，终将使训练集中的所有向量分类正确。

10.2 统计决策图像模式识别

10.2-1 基本理论

在实际图像模式识别的问题中，由于每个类别中的图像模式极不可能有单一的特征向量值，而是一段范围内的值，而每个图像模式的特征值会落在该范围的某个点并无法精准预知，甚至所谓的明确数值范围也未必可知。对于这种具有随机特性的问题，以概率的概念来描述极为适合。

设某一个图像模式 $\boldsymbol{x}$ 来自 ω_i 类的概率为 $p(\omega_i \mid \boldsymbol{x})$。另外假设一个分类器判定某个图像模式来自 ω_j，但事实上它来自 ω_i 所造成误判的损失为 L_{ij}。因此，将图像模式 $\boldsymbol{x}$ 归类为类别 ω_j 的平均损失为

$$\rho_j(\boldsymbol{x}) = \sum_{k=1}^{M} L_{kj} p(\omega_k \mid \boldsymbol{x}) \tag{10.2-1}$$

其中 M 为类别总数。上式又可改写成

$$\rho_j(\boldsymbol{x}) = \frac{1}{p(\boldsymbol{x})} \sum_{k=1}^{M} L_{kj} p(\boldsymbol{x} \mid \omega_k) P(\omega_k) \tag{10.2-2}$$

其中 $p(\boldsymbol{x} \mid \omega_k)$ 为 ω_k 类别的图像模式概率密度函数，$P(\omega_k)$ 则是出现 ω_k 类的概率。由于 (10.2-2) 式中的共同系数 $p(\boldsymbol{x})$ 与类别无关，对类别的判定无影响，因此可删除，于是 (10.2-2) 式变成

$$\rho_j(\boldsymbol{x}) = \sum_{k=1}^{M} L_{kj} p(\boldsymbol{x} \mid \omega_k) P(\omega_k) \tag{10.2-3}$$

决策时，对一输入图像模式 $\boldsymbol{x}$ 计算所有的平均损失 $\rho_1(\boldsymbol{x}), \rho_2(\boldsymbol{x}), \cdots, \rho_M(\boldsymbol{x})$，并选择具有最小平均损失的类别当做 $\boldsymbol{x}$ 的归属。此种分类器称为贝氏分类器 (Bayes classifier)。

若分类正确时损失为 0，而分类错误时的损失都是固定值 (与类别无关)，例如设为 1，则 L_{kj} 可写成 $1-\delta_{kj}$，因而 (10.2-3) 式可简化成

$$\begin{aligned}\rho_j(\boldsymbol{x}) &= \sum_{k=1}^{M} (1-\delta_{kj}) p(\boldsymbol{x} \mid \omega_k) P(\omega_k) \\ &= p(\boldsymbol{x}) - p(\boldsymbol{x} \mid \omega_j) P(\omega_j)\end{aligned} \tag{10.2-4}$$

于是贝氏分类器在下列条件下会将图像模式 $\boldsymbol{x}$ 归类为 ω_i

$$p(\boldsymbol{x}) - p(\boldsymbol{x} \mid \omega_i) P(\omega_i) < p(\boldsymbol{x}) - p(\boldsymbol{x} \mid \omega_j) P(\omega_j) \tag{10.2-5}$$

或

$$p(\boldsymbol{x} \mid \omega_i) P(\omega_i) > p(\boldsymbol{x} \mid \omega_j) P(\omega_j), \quad j = 1, 2, \cdots, M; j \neq i \tag{10.2-6}$$

因此具有 0 ~ 1 失真损失的贝氏分类器相当于采用决策函数

$$D_j(\boldsymbol{x}) = p(\boldsymbol{x} \mid \omega_j)P(\omega_j), \quad j = 1,2,\cdots,M \tag{10.2-7}$$

来做分类。

10.2-2　高斯图像模式类别的贝氏分类器

在 (10.2-7) 式中，需要 $p(\boldsymbol{x} \mid \omega_j)$，即第 j 类的 $\boldsymbol{x}$ 的概率密度分布。最常见的是高斯分布

$$p(\boldsymbol{x} \mid \omega_j) = \frac{1}{(2\pi)^{n/2} \mid \boldsymbol{C}_j \mid^{1/2}} \exp\left[-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{m}_j)^{\mathrm{T}} \boldsymbol{C}_j^{-1} (\boldsymbol{x} - \boldsymbol{m}_j)\right] \tag{10.2-8}$$

其中 n 为特征向量的维度，$\boldsymbol{m}_j$ 与 $\boldsymbol{C}_j$ 分别为第 j 类图像模式的平均向量与协方差矩阵。在此情况下，比较简便的决策函数是

$$\begin{aligned} D_j(\boldsymbol{x}) &= \ln\left[p(\boldsymbol{x} \mid \omega_j)P(\omega_j)\right] \\ &= \ln P(\omega_j) - \frac{n}{2}\ln 2\pi - \frac{1}{2}\ln \mid \boldsymbol{C}_j \mid - \frac{1}{2}\left[(\boldsymbol{x} - \boldsymbol{m}_j)^{\mathrm{T}} \boldsymbol{C}_j^{-1} (\boldsymbol{x} - \boldsymbol{m}_j)\right] \end{aligned} \tag{10.2-9}$$

其中 $\frac{n}{2}\ln 2\pi$ 与类别无关，故可省略。另外，若 $P(\omega_j) = \frac{1}{M}$，即机会均等，则 (10.2-9) 式可进一步简化成

$$D_j(\boldsymbol{x}) = -\frac{1}{2}\ln \mid \boldsymbol{C}_j \mid - \frac{1}{2}\left[(\boldsymbol{x} - \boldsymbol{m}_j)^{\mathrm{T}} \boldsymbol{C}_j^{-1} (\boldsymbol{x} - \boldsymbol{m}_j)\right], \quad j = 1,2,\cdots,M \tag{10.2-10}$$

若所有协方差矩阵又都一样，即 $\boldsymbol{C}_j = \boldsymbol{C}, j = 1,2,\cdots,M$，则可再简化成

$$D_j(\boldsymbol{x}) = \boldsymbol{x}^{\mathrm{T}} \boldsymbol{C}^{-1} \boldsymbol{m}_j - \frac{1}{2}\boldsymbol{m}_j^{\mathrm{T}} \boldsymbol{C}^{-1} \boldsymbol{m}_j, \quad j = 1,2,\cdots,M \tag{10.2-11}$$

若又进一步，$\boldsymbol{C} = \boldsymbol{I}$ (单位矩阵)，则

$$D_j(\boldsymbol{x}) = \boldsymbol{x}^{\mathrm{T}} \boldsymbol{m}_j - \frac{1}{2}\boldsymbol{m}_j^{\mathrm{T}} \boldsymbol{m}_j, \quad j = 1,2,\cdots,M \tag{10.2-12}$$

我们可证明 (10.2-12) 式为最短距离分类器 (minimum distance classifier)，此证明留做习题。所谓最短距离分类器是指计算 $\boldsymbol{x}$ 至各类别平均向量 $\boldsymbol{m}_j$ 的距离最近者将 $\boldsymbol{x}$ 归类至该类。

10.3　特征选取

在图像模式识别的问题中，我们常需要在众多可利用的特征中选取适当者以供分类器应用。特征选取的问题受到许多文献的广泛讨论，但是并没有明确的答案。

事实上，对于特征的选取并没有太多可供遵循的准则。一般来说，本质上有用的特征都是凭借直观来选取的。以下依序列举了少数较佳的特征。

1. 区别性 (discrimination)：对于属于不同分类的样本来说，其特征应含有不同且具有意义的值。以第八章图 8.1-2 所举的水果特征空间为例，直径是一个很好的特征。

2. 可靠性 (reliability)：对于所有同一分类的所有样本来说，特征应具有类似的值。例如，颜色对于分辨苹果的成熟度来说可能不尽理想，也就是说，即使一个青苹果和一个红苹果 (成熟的苹果) 属于同样的种类，但以颜色特征来说却有不同的值。

3. 独立性 (independence)：这是说所使用的特征间应该彼此互不相关。例如直径和重量就构成高度相关的特征，因为重量几乎正比于直径的立方。

4. 数目小 (small numbers)：图像模式识别系统的复杂度随其维数 (使用特征的数目) 增加而快速增加。更重要的是，训练分类器与评估其效率所需的样本数随特征数增加而呈指数增加。

如上所述，我们一直在寻找一小组可靠、独立且具辨认力的特征。一般来说，我们期望分类器的效率在有用的特征被剔除时能随之下降，以此验证某些特征的有效性。另外，去除高频噪声与高相关性的特征可以改善分类器的效率。

特征选取可以看成是去除某些特征及结合其他相关的特征，直到特征集变成易于管理而效能仍足够的过程。在以下的讨论中，我们将考虑一个将二个特征减为一个特征的简单问题。假设有一个含有 M 个类别样本的训练集，设 N_j 为第 j 个类别的样本数目，且第 i 个样本在类别 j 中做判断时所得的特征为 x_{ij} 和 y_{ij}。

接下来，我们开始计算各类别中每个特征的平均值

$$\hat{\mu}_{x_j} = \frac{1}{N_j}\sum_{i=1}^{N_j} x_{ij} \tag{10.3-1}$$

和

$$\hat{\mu}_{y_j} = \frac{1}{N_j}\sum_{i=1}^{N_j} y_{ij} \tag{10.3-2}$$

μ_{x_j} 与 μ_{y_j} 上方的 ^ 记号表示是以训练集所产生类别平均值的估测，而非真正类别的平均值。

类别 j 中特征 x 的估测方差为

$$\hat{\sigma}_{x_j}^2 = \frac{1}{N_j}\sum_{i=1}^{N_j} (x_{ij} - \hat{\mu}_{x_j})^2 \tag{10.3-3}$$

同理，特征 y 的估测方差为

$$\hat{\sigma}_{y_j}^2 = \frac{1}{N_j}\sum_{i=1}^{N_j} (y_{ij} - \hat{\mu}_{y_j})^2 \tag{10.3-4}$$

- **特征相关性**

类别 j 中特征 x 及 y 的相关性可由下式估测

$$\hat{\sigma}_{xy_j} = \frac{\frac{1}{N_j}\sum_{i=1}^{N_j}(x_{ij}-\hat{\mu}_{x_j})(y_{ij}-\hat{\mu}_{y_j})}{\hat{\sigma}_{x_j}\hat{\sigma}_{y_j}} \tag{10.3-5}$$

$\hat{\sigma}_{xy_j}$ 的值被限制在 - 1 与 1 之间，若为零值则表示二特征不相关。若为接近于 + 1 的值则表示具有高度相关性。若值为 - 1 则表示其中一个特征变量与另一特征变量的负数成比例。若这个值的大小接近 1 则表示两个特征或许可以结合为一，亦或其中之一可以去除。

● **类别相隔的距离**

好的特征可使类别间容易被区分，一个衡量特征区分两个类别能力的指标称为经方差归一化后的分类平均值间的距离。以特征 x 来说，其定义如下

$$\hat{D}_{x_{jk}} = \frac{\left|\hat{\mu}_{x_j} - \hat{\mu}_{x_k}\right|}{\sqrt{\hat{\sigma}_{x_j}^2 + \hat{\sigma}_{x_k}^2}} \tag{10.3-6}$$

此处两个分类分别为 j 和 k，显然此距离愈大愈好。

● **降低维度**

有许多方法可将特征 x 和 y 结合为单一特征 z，一个简单方法是将 x 与 y 以某个比例加以组合的线性函数

$$z = ax + by \tag{10.3-7}$$

因为调整特征值的大小不会影响分类器的效率 (每个输入图像模式都做相同的调整)，我们可以对调整比例加上限制，例如

$$a^2 + b^2 = 1 \tag{10.3-8}$$

将上式并入 (10.3-7) 式，写成

$$z = x\cos\theta + y\sin\theta \tag{10.3-9}$$

此处 θ 标示出 x 与 y 在 z 中比例的新变量。使用时可选取 θ 使得特征 z 有最大的类别区分能力 [以 (10.3-6) 式或其他方式衡量]。

10.4 聚　类

10.4-1　聚类的定义

聚类 (clustering) 的定义为对所给予的特征空间内的各点，将其分成若干类的分割处理。根据图像模式的识别目的，聚类有时不必用到复杂的算法。例如，对印刷后的数字

{1,2,…,9,0} 做识别时，其类别数 10 个为已知，因此只要以预先储存的标准图像模式来与输入的图像模式做匹配即可。而以手写的 10 种数字{1,2,…,9,0}做识别时亦相同。首先其类别数 10 个为已知，利用训练学习所用的图像模式，在这些图像模式代表某个数字可由人判断的情况下，找出特征空间中各个数字存在的范围及有系统的查验法则。而后只要检查未知图像模式在这些标准空间属何范围，即可做图像模式的辨认，此时亦不必用到复杂的聚类方法。然而，在以下情况，应用以上所提出的简单方式并不适合：

① 辨认对象数为未知。

② 未给予标准图像模式。

③ 同一类中差别较大，在标准空间中，其输入的图像模式做单纯的比较困难者。

在这些情形下要用到聚类的方法。此方法将多个图像模式相似者分别做汇集，并分成几个群。此种处理过程在图像模式识别上是非常重要的课题。

聚类算法要处理以下两个重要的问题：

① 聚类总数的决定。

② 聚类群的生成。

考虑图 10.4-1 (a) 所示的情形，如图中所示，很明显可分成 2 群。接着考虑图 10.4-1 (b) 情形，若以图中虚线的方式聚类，则可用 1 群来包含所有点，若是以图中实线的方式聚类，则可区分成 3 群，至于要分成多少群则要根据图像模式间的相似性做判断。而于 3 个群的情形时，对图中的 $\boldsymbol{x}$ 而言，因其与 3 个群皆有距离，此时其属何群即成问题。此种聚类问题是较难的问题。

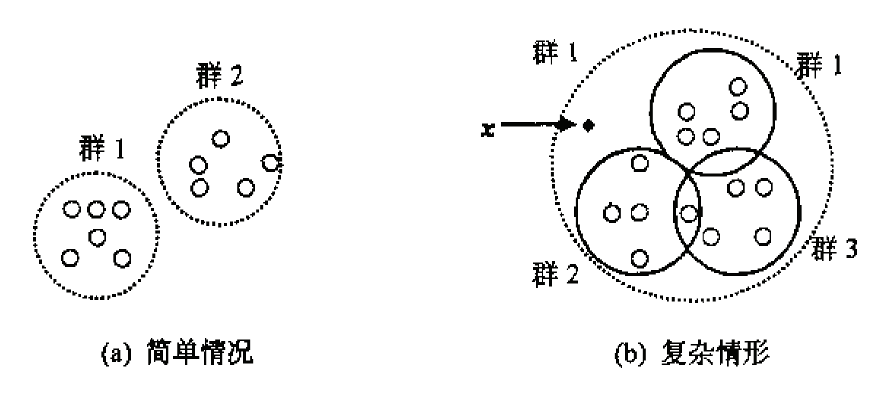


图 10.4-1 图像模式聚类案例

10.4-2 聚类算法

本节将介绍一些聚类算法。设有 N 个图像模式的特征向量 $\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_N$ 且于群 ω_i $(i=1,2,\cdots)$ 中二个图像模式间的距离定义如下：

$$(\text{图像模式 } \boldsymbol{x}_i \text{ 及 } \boldsymbol{x}_j \text{ 间的距离}) \equiv d(\boldsymbol{x}_i,\boldsymbol{x}_j) \tag{10.4-1}$$

而某一图像模式 $\boldsymbol{x}_i$ 与某群 ω_i 之间的距离，则设为 $\boldsymbol{x}_i$ 与 ω_i 标准图像模式间的距离，并表示成

$$(\text{图像模式 } \boldsymbol{x}_i \text{ 与群 } \omega_i \text{ 间的距离}) \equiv d(\boldsymbol{x}_i,\omega_i) \tag{10.4-2}$$

● **最邻近** (Nearest Neighbor, NN) 法

NN 法的步骤如下：

(1) 设处理时的图像模式编号为常数 P，其初始值 $P=1$。

所形成的群数设为常数 M，其初始值为 $M=1$。

设 $\boldsymbol{x}_P$ 为标准图像模式，形成初始群 ω_M。设 T 为一个小门限值。

(2) 求距离

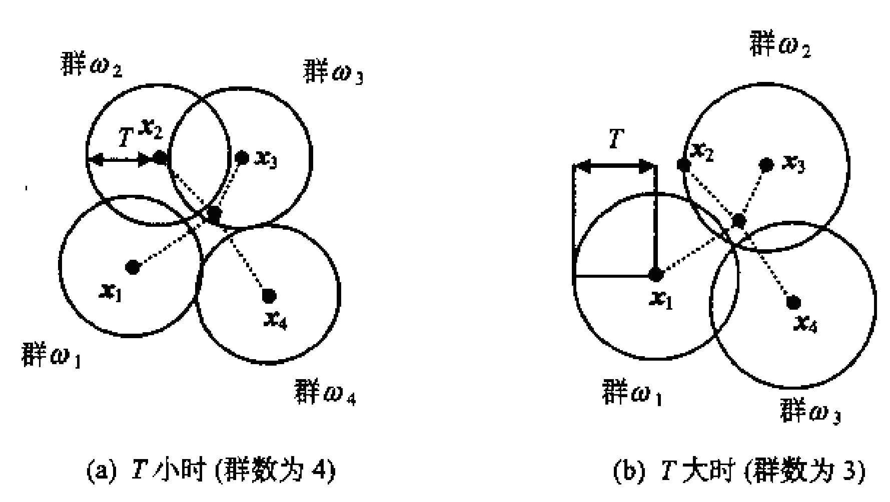
$$d_{\min}=\min_{1\leq i\leq M}\{d(\boldsymbol{x}_{P+1},\omega_i)\} \tag{10.4-3}$$

当 $d_{\min}<T$ 时，$\boldsymbol{x}_{P+1}$ 属于具有 $d_{\min}$ 的 ω_i。

当 $d_{\min}\geqslant T$ 时，$\boldsymbol{x}_{P+1}$ 当作标准图像模式，而形成新群 ω_{M+1}，并对 M 值加 1。P 值也加 1。

(3) $P>N$ 时，执行结束；否则回到 (2)。

门限值 T 的大小对 NN 法聚类的结果会有很大的影响。如图 10.4-2 (a) 所示，整体生成 4 个群，而在门限值设大时，则如图 10.4-2 (b) 所示，所生成的群数减至 3 个。



(a) T 小时 (群数为 4)　　(b) T 大时 (群数为 3)

图 10.4-2　在 NN 法中门限值 T 的影响

● ***K* 最邻近** (*K* Nearest Neighbor, *K*-NN) 法

K-NN 法与 NN 法类似，其步骤如下：

(1) 与 NN 法相同。

(2) 求距离 $d_j\ (1\leqslant j\leqslant M)$：

$$d_j=d(\boldsymbol{x}_{P+1},\omega_j) \tag{10.4-4}$$

在 d_j 比门限值 T 小时，依小到大排序后，至 K 号群为止，将图像模式 $\boldsymbol{x}_{P+1}$ 重复归属于上述 K 个群。P 值加 1。

(3) 与 NN 法相同。

在 *K*-NN 法中，某图像模式 $\boldsymbol{x}$ 归属某群情形则如图 10.4-3 所示。如图中所示，在 $K=3$ 时，$\boldsymbol{x}$ 重复归属于 $\omega_\alpha,\omega_\beta$ 与 ω_γ 三群。因 *K*-NN 法与 NN 法相同，亦受门限值 T 的大小所影响，故必需要多试几个 T 来做聚类，然后采用最适合者做处理。

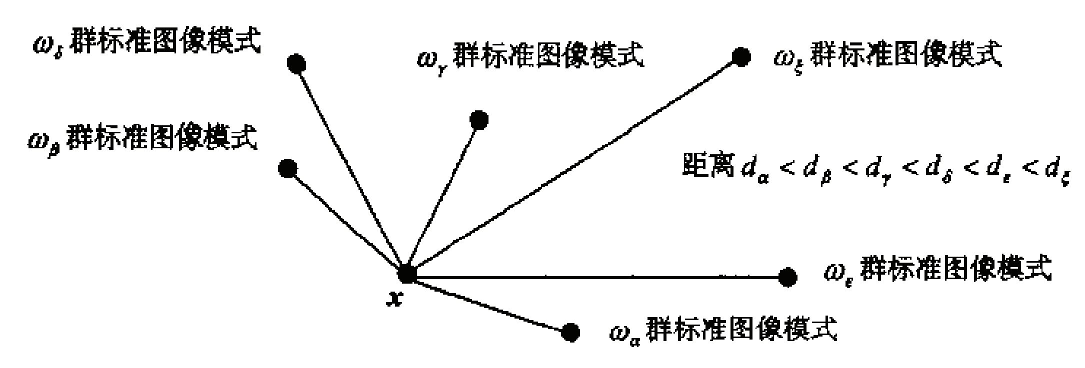


图 10.4-3　K-NN 法中图像模式的重复登录

● **K 平均 (K-mean) 算法**

此为将群数固定为 K 个而做聚类的算法。其步骤如下：

(1) 选定 K 个适当的图像模式，作为群 $\omega_1 \sim \omega_K$ 的标准图像模式。令

$$P=1, \qquad M=K$$

(2) 当下式成立时，将图像模式 $\boldsymbol{x}_P$ 归属于群 ω_j

$$d(\boldsymbol{x}_P,\omega_j) < d(\boldsymbol{x}_P,\omega_i) \tag{10.4-5}$$

此处 $i=1,2,\cdots,K$； $i \neq j$。

P 值增加 1。当 $P>N$ 时，设 $P=1$。

(3) 群 ω_j 的新标准图像模式 $\boldsymbol{m}_j$ 是使下式中所示的成本函数 J 为最小的情况下所定。

$$J=\sum_{x_i \in C_j}\left|d(\boldsymbol{m}_j,\boldsymbol{x}_i)\right|^2 \tag{10.4-6}$$

此时的新标准图像模式 $\boldsymbol{m}_j$，为属于群 ω_j 的全体图像模式 $\boldsymbol{x}_i$ $(\in \omega_j)$ 的平均

$$\boldsymbol{m}_j=\frac{1}{N_j}\sum_{x_i \in \omega_j}\boldsymbol{x}_i \tag{10.4-7}$$

N_j 为属 ω_j 的图像模式总数。

(4) 对所有的群以 (3) 求其新标准图像模式，并在与前次标准图像模式相等时，终止处理。否则返回 (2)。

以 K-平均算法，对某群 ω_i 的标准图像模式做更新的状态如图 10.4-4 所示。以此方法所产生群的个数固定为 K 个。应多尝试不同的 K 值，以选出最适合的 K 值。

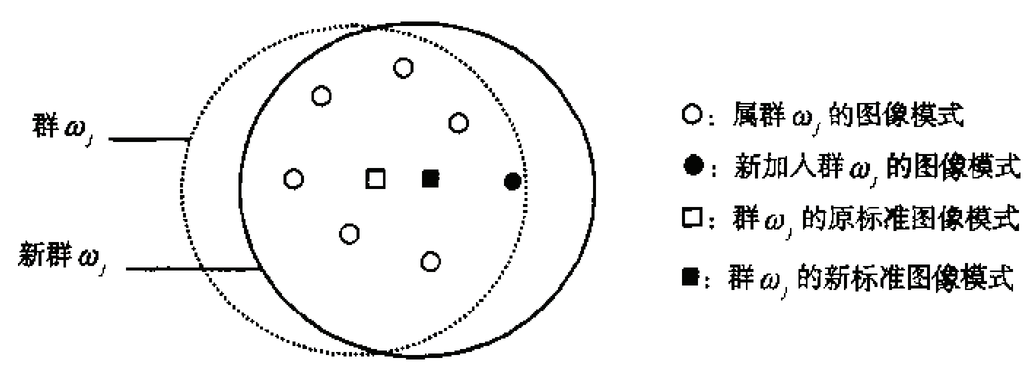


图 10.4-4　K-平均算法中标准图像模式的更新

10.4-3 模糊聚类

在先前讨论的聚类观念中，一个图像模式会很明确地被指定属于某一类，不会同时归属于两个或两个以上的类别。但是我们也发现在许多问题中，类别的边界并不是永远都是如此清晰分明，也就是有所谓的灰色地带，模糊聚类 (fuzzy clustering) 便是针对这种观念所提出的聚类方式。在模糊聚类中，每一个图像模式归属于某一类是以其 0 到 1 的数字代表其归属程度，其中 1 与 0 分别代表最高与最低的归属度。

模糊聚类的最后结果是以一个分割矩阵 (partition matrix) $\boldsymbol{U}$ 来呈现

$$\boldsymbol{U}=\{u_{ij}\},\quad i=1,2,\cdots,M \quad j=1,2,\cdots,N \tag{10.4-8}$$

其中 $u_{i,j}\in[0,1]$，代表图像模式 $\boldsymbol{x}_j$ 归属于第 i 个类别的程度。此外有两个伴随 u_{ij} 的天然限制，首先是

$$\sum_{i=1}^{M}u_{ij}=1,\quad 对所有\ j=1,2,\cdots,N \tag{10.4-9}$$

另外一个是

$$0<\sum_{j=1}^{N}u_{ij}<N,\quad 对所有\ i=1,2,\cdots,M \tag{10.4-10}$$

在上一节的 K-平均 (又称 C-平均) 聚类准则中，是以将 (10.4-6) 式的成本函数最小化的方式获得群心位置，此处则可用如下的成本函数以及 (10.4-9) 及 (10.4-10) 的两个条件限制来求

$$J(u_{ij},\boldsymbol{m}_i)=\sum_{i=1}^{M}\sum_{j=1}^{N}u_{ij}^{r}\,|\boldsymbol{x}_j-\boldsymbol{m}_i|^2,\quad r>1 \tag{10.4-11}$$

其中 $\boldsymbol{m}_i$ 为第 i 群的群心，r 称为指数型权重，此权重影响分割矩阵归属值的模糊化程度。上述的问题是一个典型的有条件限制的最佳化问题，解得

$$\boldsymbol{m}_i=\frac{1}{\sum_{j=1}^{N}(u_{ij})^r}\sum_{j=1}^{N}(u_{ij})^r\boldsymbol{x}_j,\quad i=1,2,\cdots,M \tag{10.4-12}$$

$$u_{ij}=\frac{\left(1/|\boldsymbol{x}_j-\boldsymbol{m}_i|^2\right)^{\frac{1}{r-1}}}{\sum_{k=1}^{M}\left(1/|\boldsymbol{x}_j-\boldsymbol{m}_k|^2\right)^{\frac{1}{r-1}}},\quad i=1,2,\cdots,M;\ j=1,2,\cdots,N \tag{10.4-13}$$

以上的结果是由 Bezdek[1981]所提出的模糊 C-平均 (fuzzy C-mean, FCM) 版本。整个算法的步骤如下：

(1) 选取群数 M $(2\le M\le N)$ 以及指数型权重 r $(1<r<\infty)$。选取一个起始的分割矩阵 $\boldsymbol{U}^{(0)}$ 以及结束程序所需的误差临界值 ε。设循环的次数 t 为 0。

(2) 用$\boldsymbol{U}^{(t)}$及 (10.4-12) 式计算模糊聚类中心$\left\{\boldsymbol{m}_i^{(t)} \mid i=1,2,\cdots,M\right\}$。

(3) 用$\left\{\boldsymbol{m}_i^{(t)} \mid i=1,2,\cdots,M\right\}$及 (10.4-13) 式计算新的分割矩阵$\boldsymbol{U}^{(t+1)}$。

(4) 计算$\Delta=\left|\boldsymbol{U}^{(t+1)}-\boldsymbol{U}^{(t)}\right|=\max\limits_{i,j}\left|u_{ij}^{(t+1)}-u_{ij}^{(t)}\right|$。若$\Delta>\varepsilon$，则$t=t+1$，回到 (2)；否则结束。

10.5 利用人工神经网络做图像模式识别

1957 年 Rosenblatt 提出感知器 (perceptron) 模型。当时、以感知器为中心、对人工神经网络的研究曾盛极一时。然而，感知器网络受限于当时的人工神经网络无法以硬件来实现，曾导致人工神经网络的研究停滞不前。另一个受挫的关键是 Minsky 与 Papert 写了一本叫感知器的书，书中以数学证明，当时人工神经网络的学习能力极差、甚至连最简单的“异或”(exclusive or, XOR) 的问题都无法解决。

直到 1982 年 Hopfield 提出 Hopfield 人工神经网络模型，1986 年 Rumelhart 等人提出反向传播 (backpropagation, BP) 学习法，及计算机处理能力的长足进步，人工神经网络的研究才再次受到重视。本节将介绍人工神经网络的基本概念及其模型。

10.5-1 人工神经细胞模型

人工神经网络的神经细胞模型如图 10.5-1 所示，可视为具多输入及单一输出的信息处理单元。它是人工神经网络构成的基本单位，称为神经元。神经元会随输入的不同而使内部状态变化，并根据内部状态来输出。由多个神经元所结合成的网络，即为人工神经网络。

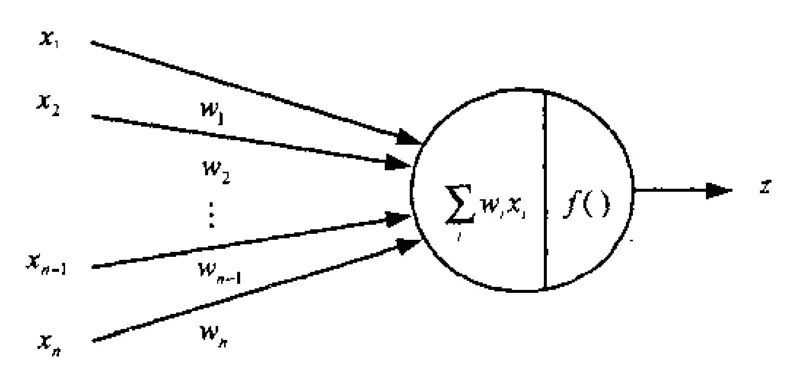


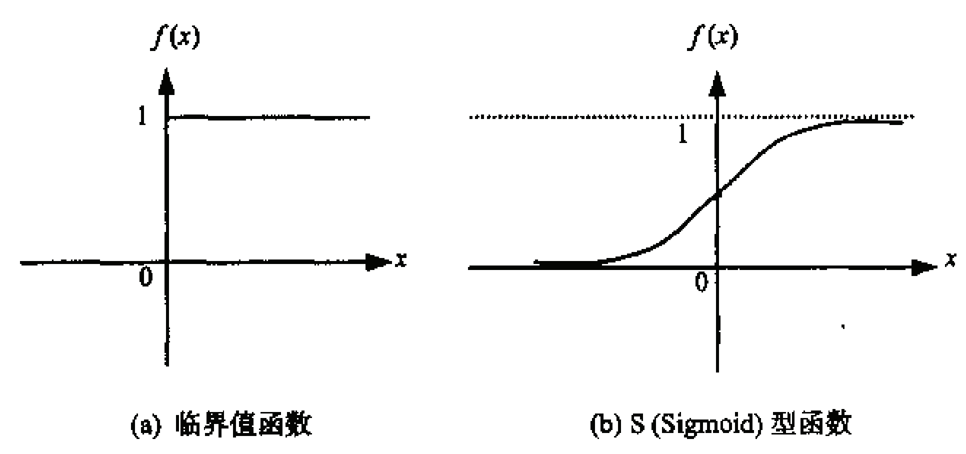
图 10.5-1 人工神经细胞模型

一般神经元间的结合可配以加权值，而随加权值的变化，网络便可调整到所希望的输出。在图 10.5-1 中，某个神经元的输人是由其他神经元的输出值x_i乘上加权值w_i的总和值$\sum_i w_i x_i$。而神经元则以此总和值输入一转移函数$f()$之后的值作为输出。常用的两个转移函数$f()$如下，其图形则如图 10.5-2 所示。

$$f(x)=\begin{cases}1, & x\geqslant 0\\ 0, & x<0\end{cases} \tag{10.5-1}$$

$$f(x)=\frac{1}{1+\exp(-x/a)} \tag{10.5-2}$$

其中 a 为常数。

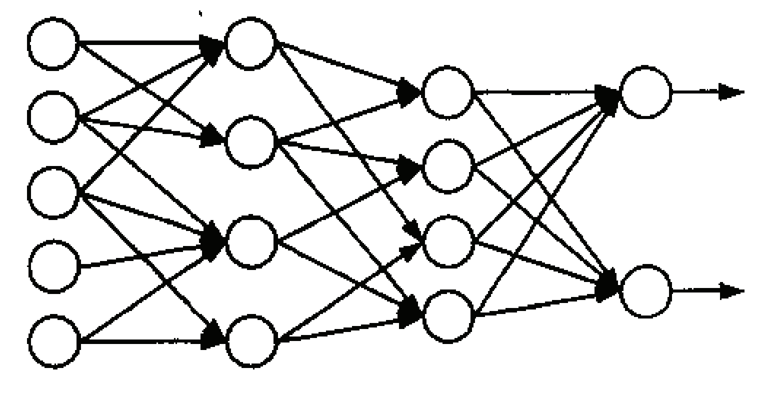


(a) 临界值函数　　　　(b) S (Sigmoid) 型函数

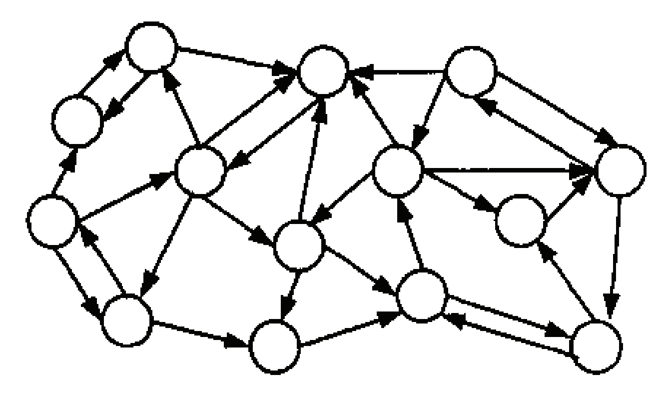
图 10.5-2　转移函数实例

10.5-2　网络的结构及学习方法

人工神经网络可根据网络结构及学习方法加以区分，其中在网络结构上，则可分成阶层型及相互结合型，如图 10.5-3 所示。



(a) 阶层型网络



(b) 相互结合型网络

图 10.5-3　网络结构

● **阶层型**

此型的神经元以层状配置，信号由输入层传达到输出层，而以层跟层来结合成网络。一般以由某层的某神经元开始，与次层的所有神经元结合。此外，上层与下层之间的结合亦有以反馈来设定者。感知器为阶层型网络的例子之一。

● **相互结合型**

相互结合型并没有层的存在，而是以任意神经元做两个方向的结合而成的网络。典型的例子则有 Hopfield 模型等。

决定网络结构之后，下一步是以人工神经网络来执行信息的处理。一般都需要先做网络的学习。所谓网络的学习是指将加权值调整到适应训练集为止。而阶层型网络的有效学习方式之一是先前提到的反向传播学习算法，我们将在 10.5-3 节介绍。另外也有以网络的平衡状态作为信息处理结果的网络，如 10.5-4 节的 Hopfield 模型等。

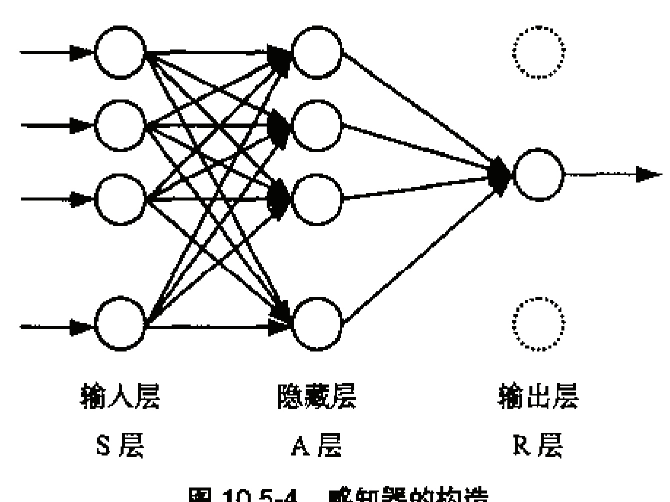


图 10.5-4 感知器的构造

10.5-3 感知器及反向传播学习算法

● **感知器**

Rosenblatt 为做图像模式分类所提出的感知器原型由 S (Sensory) 层、A (Association) 层及 R (Response) 层所构成，其中同一层的神经元不相互结合，而层与层间的结合则为由 S 层至 R 层的单方向。在 S 层与 A 层间的连结是固定的，而 A 层至 R 层间的连结则为可变的。除输入层外，各层神经元的输入为前一层神经元的输出乘以加权值而来，再经神经元的转移函数处理后，输出到下一层。

为方便说明起见，定义下列参数：

j：第 $k-1$ 层神经元的编号。

i：第 k 层神经元的编号。

p：第 $k+1$ 层神经元的编号。

m：表示输出层号码。

θ_i^k：第 k 层第 i 个神经元的临界值。

I_i^k：第 k 层第 i 个神经元的输入减掉临界值的净值。

O_i^k：第 k 层第 i 个神经元的输出。

$w_{j,i}^{k-1}$：由第 $k-1$ 层之第 j 个神经元至第 k 层之第 i 个神经元的加权值。

$f()$：转移函数。

上述参数的关系如图 10.5-5 所示。

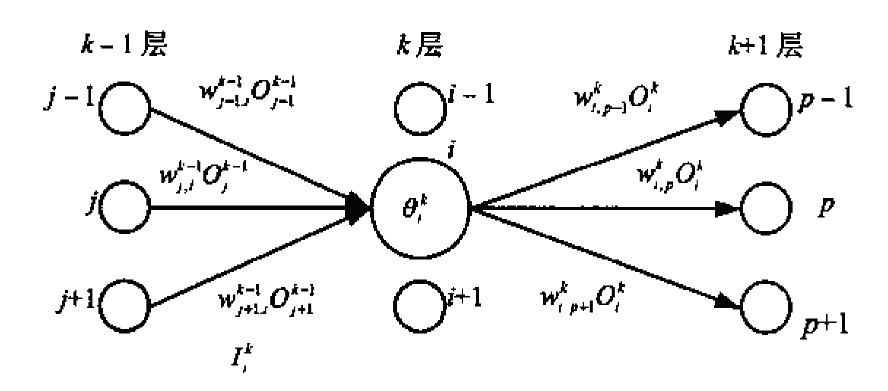


图 10.5-5　网络参数间的关系

若以数学式表之则有

$$I_i^k = \sum_j w_{j,i}^{k-1} \cdot O_j^{k-1} - \theta_i^k \tag{10.5-3}$$

$$O_i^k = f(I_i^k) \tag{10.5-4}$$

● **反向传播学习算法**

反向传播学习算法是阶层型网络中一个有效的学习算法，广被采用，也是目前感知器的典型学习算法。

反向传播学习是利用最优化理论中的梯度下降 (gradient steepest descent) 法的观念，将所需要的输出与实际输出间的误差逐步最小化的一个方法。此输出误差往输入方向“反向传播”回去，以便调整加权值将误差减少，这是称为反向传播的原因。以下将推导加权值 $w_{j,i}^k$ 所需的修正量 $\Delta w_{j,i}^k$。

反向传播学习是监督式学习的一种，亦即对应一个输入有预期的输出，观察实际输出与预期输出之差来监控网络的学习。此种网络学习模式一般以下列的能量函数 (或称为误差函数) 来表示学习的品质

$$E = \frac{1}{2}\sum (O_i^m - y_i)^2 \tag{10.5-5}$$

其中 y_i 为输出层第 i 个神经元的预期输出，O_i^m 则为输出层第 i 个神经元的实际输出。上式中取 1/2 是为了将来对 E 微分时可获得较简洁的结果。要使 E 减少所需的加权值修正量应与 E 对加权值的偏微分成正比，亦即

$$\Delta w_{j,i}^k = -\alpha \frac{\partial E}{\partial w_{j,i}^k} \tag{10.5-6}$$

其中α为一正值的修正比例常数。上式又可写成

$$\Delta w_{j,i}^k = -\alpha \frac{\partial E}{\partial I_i^{k+1}} \frac{\partial I_i^{k+1}}{\partial w_{j,i}^k} = -\alpha \frac{\partial E}{\partial I_i^{k+1}} O_j^k = -\alpha d_i^{k+1} O_j^k \tag{10.5-7}$$

此处令$\frac{\partial E}{\partial I_i^{k+1}} = d_i^{k+1}$。

在$k+1 \neq m$时，设转移函数f的导数为f'，则d_i^{k+1}可以写成

$$d_i^{k+1} = \frac{\partial E}{\partial I_i^{k+1}} = \sum_p \frac{\partial E}{\partial I_p^{k+1}} \frac{\partial I_p^{k+1}}{\partial O_i^k} \frac{\partial O_i^k}{\partial I_i^k} = \sum_p \frac{\partial E}{\partial I_p^{k+1}} w_{i,p}^k f'(I_i^k) \tag{10.5-8}$$

在$k+1 = m$时，对第m层的第i个神经元，以y_i当其监督信号，则d_i^{k+1}可以写成

$$d_i^m = (O_i^m - y_i) f'(I_i^m) \tag{10.5-9}$$

综合上述结果可得

$$\Delta w_{j,i}^k = -\alpha d_i^{k+1} O_j^k \tag{10.5-10}$$

其中

$$d_i^m = (O_i^m - y_i) f'(I_i^m) \tag{10.5-11}$$

$$d_i^{k+1} = \left[\sum_p w_{i,p}^k d_p^{k+1}\right] f'(I_i^k), \quad k+1 < m \tag{10.5-12}$$

若转移函数为 (10.5-2) 式中的 S 型函数时，则下式成立

$$f'(I_i^k) = f(I_i^k)\left[1 - f(I_i^k)\right] = O_i^k (1 - O_i^k) \tag{10.5-13}$$

反向传播学习算法有几个待决定的重要参数，包括隐藏 (非输入与输出层) 的层数、每个隐藏层的神经元及学习速度。一般而言用 1 到 2 层的隐藏层可得不错的结果，这是因为没有隐藏层，不能反应问题输入各种神经元间的交互作用，误差较大；过多的隐藏层又造成过多的局部最小值，使得网络容易掉入局部最小值而无法收敛。一般而言隐藏层的神经元数是取输入层与输出层的神经元数的算术平均或几何平均。此外若问题的噪声大 (方差大)，则隐藏层神经元数宜减少；问题复杂性高，隐藏层神经元可增加。学习速度快代表有较大的网络加权值修正量，有可能较快收敛，当然也有可能反覆振荡反而更难收敛。因此学习速度必须慎选之，一般取 0.5 或 0.1 ~ 1.0 之间的值都有不错的结果。

● **以反向传播网络做模糊分类**

以传统反向传播网络分类时每个图像模式一定都有一个惟一的归属类别。但在实际的问

题中，数据的归属常因没有明确的边界而难以确定。因此将模糊化的观念加入反向传播网络的训练过程中。

考虑一个 M 类的辨认问题，因而网络输出层有 M 个节点。设 n 维的向量 $\boldsymbol{m}_i$ 与 σ_i 分别代表第 i 类 ω_i 中训练集的平均值与标准差向量 。首先定义第 i 类训练模式向量与第 i 类间的加权距离为

$$z_{ji}=\sqrt{\sum_{k=1}^{n}\left(\frac{x_{jk}-m_{ik}}{\sigma_{ik}}\right)^2},\quad i=1,2,\cdots,M \tag{10.5-14}$$

其中 x_{jk} 为第 j 个模式向量的第 k 个分量，m_{ik} 与 σ_{ik} 分别为 $\boldsymbol{m}_i$ 与 σ_i 的第 k 个分量。$1/\sigma_{ik}$ 是方差归一化的因子，使得高方差者在类别的辨别上有较小的权重 (不大重要)。接着我们定义第 j 个模式在第 i 个类别 ω_i 中的归属程度为

$$u_{i,j}=\frac{1}{1+\left(z_{ji}/\alpha\right)^{\beta}},\quad i=1,2,\cdots,M \tag{10.5-15}$$

其中 α 与 β 为控制模糊度的正值参数。由上式可看出，一个图像模式与一个类别的距离愈远则其归属于该类别的程度愈低。若所有 $u_{i,j}$ 均不为零,$i=1,2,\cdots,M$ ，则有高模糊性的状况，若只有一个不为零，则完全没有模糊性。在高模糊性的状况，我们采用下面的模糊度修正子 INT 来扩大归属度的差异，以降低决策时的混淆。

$$u_{i,j,\mathrm{INT}}=\begin{cases}2\left(u_{i,j}\right)^2 & 0\leq u_{i,j}\leq 0.5\\ 1-2\left(1-u_{i,j}\right)^2 & \text{其他情况}\end{cases} \tag{10.5-16}$$

对于第 j 个图像模式 $\boldsymbol{x}_j$ ，其所预期的输出 $\boldsymbol{y}_j$ 的第 i 个分量定义为

$$y_{i,j,\mathrm{INT}}=\begin{cases}u_{i,j,\mathrm{INT}} & \text{高模糊性时}\\ u_{i,j} & \text{其他情况}\end{cases} \tag{10.5-17}$$

其中 $0\leq y_{i,j}\leq 1$，对所有 j。有这些输入与预期输出的向量对 $(\boldsymbol{x}_j,\boldsymbol{y}_j)$ ，就可以用传统的反向传播网络加以训练了。

10.5-4　Hopfield 模型

这是相互结合型人工神经网络的代表之一，其结构如图 10.5-6 所示。由网络结构中可看出，每一个神经元都跟别的神经元互有连接，但跟自己不相连，因此

$$w_{i,j}=w_{j,i} \tag{10.5-18}$$

$$w_{i,i}=0 \tag{10.5-19}$$

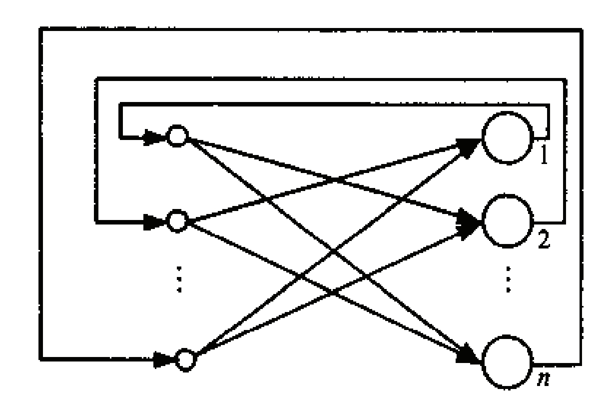


图 10.5-6 Hopfield 网络结构

Hopfield 网络是一种联想式学习网络模式，它由问题领域中取得训练集 (在此即是储存在每个神经元的状态变量值)，并从中学习训练集的内在记忆规则，以便从不完整 (不标准)的状态变量值中推论出 (联想起) 完整的或标准的状态变量值。设第 i 个神经元储存的值为 u_i，u_i = 0 或 1，则所有神经元的储存值就形成一个状态向量 $(u_1, u_1, \cdots, u_n)$。图 10.5-7 (a)显示一个具有 3 个神经元的 Hopfield 网络结构。假设每次只能改变一个神经元的储存内容，则图 10.5-7 (b) 显示所有可能的状态变化。

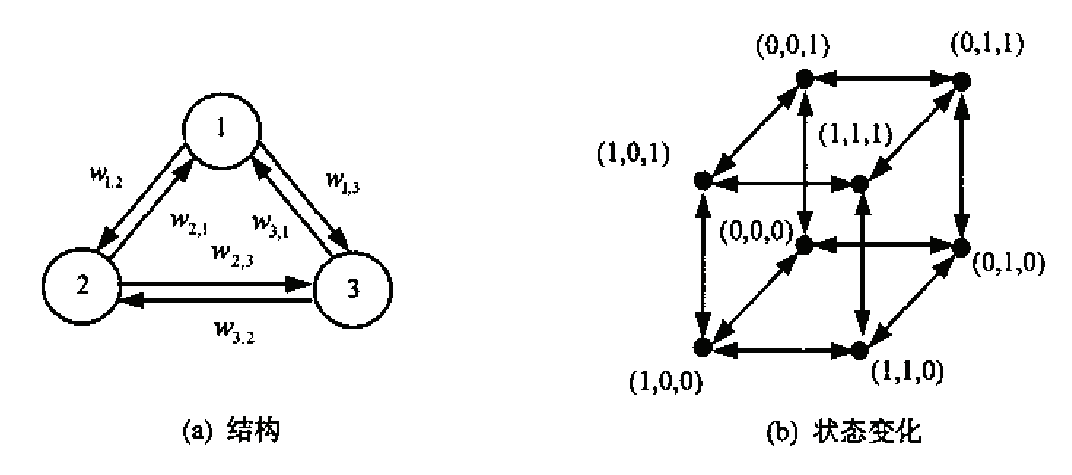


图 10.5-7 具有 3 个神经元的 Hopfield 网络的结构及其状态变化图

Hopfield 网络解决联想式学习的概念是将该问题转化成对下列能量函数 (或称 Liapunov 函数) 最小化的最优化问题

$$E = -\frac{1}{2}\sum_{j}\sum_{\substack{i \\ i \neq j}} w_{j,i} u_j u_i \tag{10.5-20}$$

当此能量函数达到最小值且每个神经元的内容都不再改变时，表示网络已达收敛状态或是训练完成。

设进入第 j 个神经元的激发值为 S_j

$$S_j = \sum_{\substack{i=1 \\ i \neq j}}^{n} u_i w_{j,i} \tag{10.5-21}$$

其对应的输出为

$$u_j=\begin{cases}1, & S_j\geqslant 0\\0, & S_j<0\end{cases} \tag{10.5-22}$$

现在证明依照上述的网络更新方法，保证网络会收敛。首先由单一个神经元 (设为第 j 个) 所贡献的能量 E_j 为

$$E_j=-\frac{1}{2}\sum_{\substack{i\\i\neq j}}w_{i,j}u_iu_j=u_j\left(-\frac{1}{2}\sum_{\substack{i\\i\neq j}}w_{i,j}u_i\right)=-\frac{1}{2}S_ju_j \tag{10.5-23}$$

因此当第 j 个神经元从先前的值 u_j^{old} 改变成现在的值 u_j^{new} 时，其能量的变化 ΔE_j 可写成

$$\Delta E_j=\left(u_j^{\text{new}}-u_j^{\text{old}}\right)\left(-\frac{1}{2}S_j\right)=-\frac{1}{2}\Delta u_jS_j \tag{10.5-24}$$

其中 Δu_j 代表 u_j 的变化。我们可轻松证明 (留做习题) $\Delta E_j\leqslant 0$，因而得证。

假设现在要网络记住 m 个状态向量 $\boldsymbol{V}_p=(v_{p1},v_{p2},\cdots,v_{pm})$，$1\leqslant p\leqslant m$。一个网络加权值的产生方式如下

$$\left.\begin{aligned}&w_{j,i}=\sum_{p=1}^{m}(2v_{p,i}-1)(2v_{p,j}-1)\quad (i\neq j)\\&w_{i,i}=0,\quad w_{i,j}=w_{j,i}\end{aligned}\right\} \tag{10.5-25}$$

考虑一个有 4 个神经元的 Hopfield 网络，则由(10.5-25) 式以及 $\boldsymbol{V}_1=(1,0,1,0)$, $\boldsymbol{V}_2=(1,0,1,0)$ 可得出其加权矩阵 $\boldsymbol{W}=\{w_{i,j}\}$

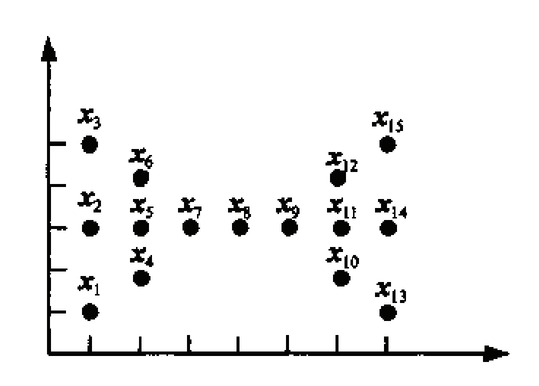
$$\boldsymbol{W}=\begin{bmatrix}0&-2&2&-2\\-2&0&-2&2\\2&-2&0&-2\\-2&2&-2&0\end{bmatrix} \tag{10.5-26}$$

假设有一输入向量 (1,1,1,0) (相当于初始状态)，则经由 (10.5-21) 与 (10.5-22) 式的过程可得最后网络的状态为 (1,0,1,0)。由此看出一个不完整或受噪声影响的图像模式，经 Hopfield 网络的联想发现是记忆中的某个图像模式，而达到辨认的目的。

习　题

1. 假设有 ω_1 与 ω_2 两个类别。ω_1 的训练集为 $\boldsymbol{x}_1=(-1,0)$, $\boldsymbol{x}_2=(0,1)$; ω_2 的训练集为 $\boldsymbol{x}_3=(0,-2)$, $\boldsymbol{x}_4=(2,0)$。以 (10.1-3) 式找出决策函数 $D(\boldsymbol{x})$，依此决策决定 $\boldsymbol{x}=(1.5,-1.5)$ 的类别归属。
2. 证明 (10.2-12) 式的决策函数形成最短距离分类器。

3. 对一个 128 × 128 大小的图像，取其不重叠的 4 × 4 子图像为聚类的输入向量 (16 维的向量)，共有多少个向量？写程序执行聚类法，分别求 4、8 及 16 类的聚类结果，包括每个类别的聚类中心及用聚类中心取代真实图像向量所得的图像，最后计算并比较这些新图像的 PSNR 值。
4. 以分成两类的 FCM 对下图所示的数据执行分类：

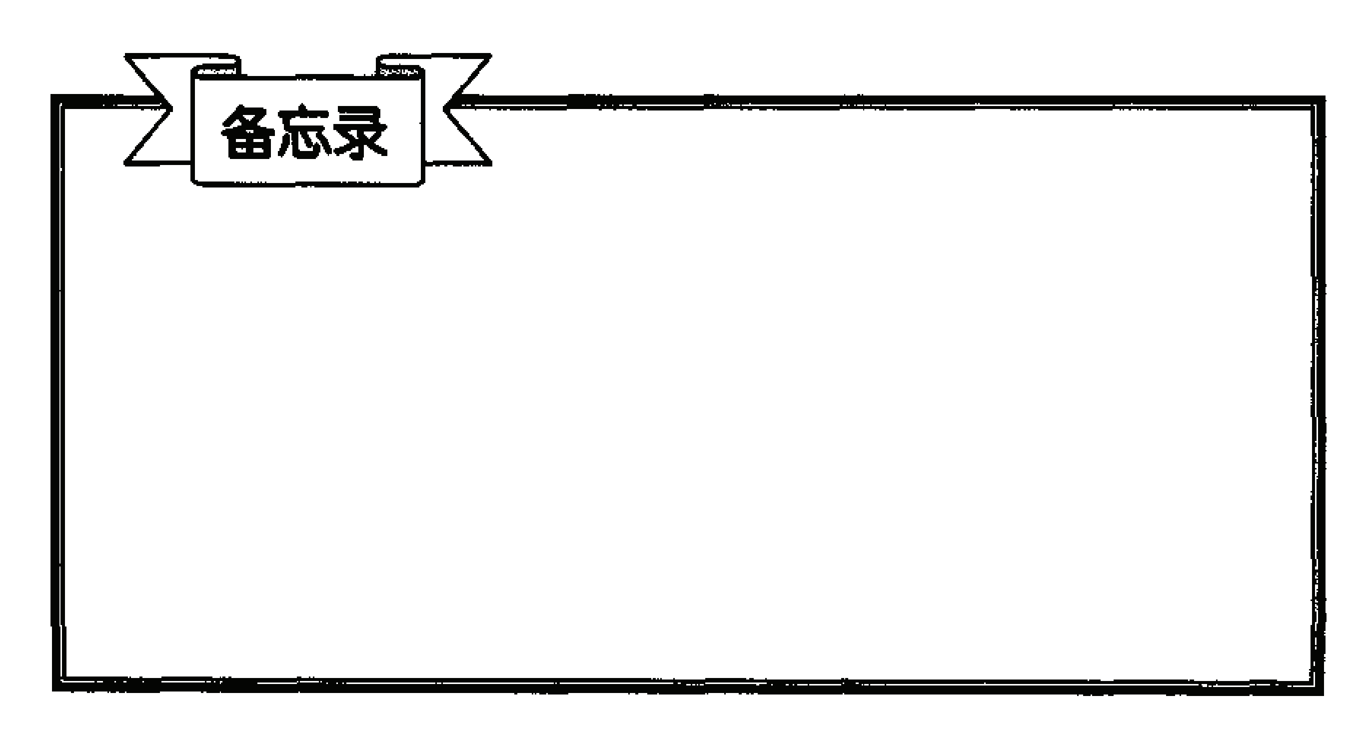


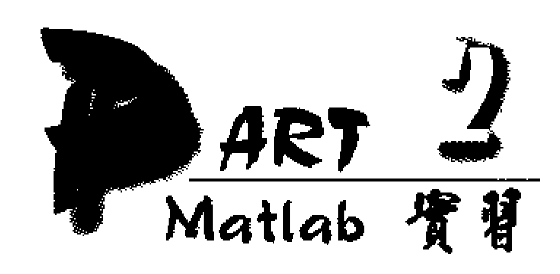
设初始分割矩阵为

$$U^{(0)}=\begin{bmatrix}0.854 & 0.146 & 0.854 & 0.854 & \cdots & 0.854\\ 0.146 & 0.854 & 0.146 & 0.146 & \cdots & 0.146\end{bmatrix}_{2\times15}$$

设误差临界值 ε 为 0.01，r 则选为 1.25。

5. 证明 (10.5-13) 式。
6. 仿图 10.5-7，画出 4 个神经元的 Hopfield 网络的结构及其状态变化图。
7. 证明 (10.5-24) 式中所定义的 ΔE_j 不为正值，即 $\Delta E_j \leqslant 0$。
8. 验证 (10.5-26) 式，同时将输入向量为 (1,1,1,0) 时获得输出状态为 (1,0,1,0) 的过程全部列出。




PART 2
Matlab 實習

第一章 Matlab 实验常用函数简介

本章介绍本书各 Matlab 实验中常用的指令及函数。若要得到详细的说明可以参考其他 Matlab 的书籍或是进入 Matlab 主程序中键入 help xxxx (xxxx 为函数或指令名称)。其中有些函数在 Matlab 的 Image 及 Wavelet toolbox 中才有。

abs	取实数的绝对值或复变量的大小
bmpread	读取.bmp 的图像文件
ceil	求不小于某数的最小整数
clear	从存储器中清除变量以及函数
conv2	做二维卷积
cos	余弦函数
cov	取方差或协方差矩阵
dct2	做二维离散余弦变换
det	行列式的值
dilate	对二值图像做膨胀
double	变为 double 精度
dwtper2	周期性单层离散二维小波变换函数
edge	边缘抽取
erode	对二值图像做腐蚀
exp	指数函数
fft2	二维快速离散傅立叶变换
figure	产生一个绘图窗口
filter2	二维数字滤波器
find	寻找非零元素的指标位置
fix	求不大于某数的最大整数
full	将稀疏矩阵变成满 (full) 矩阵
help	help 功能
histeq	执行直方图均衡化
idct2	做二维反离散余弦变换
idwtper2	周期性单层离散二维反小波变换函数
ifft2	二维快速反离散傅立叶变换
imag	取复数的虚部
imhist	显示图像的直方图
imnoise	对图像加入噪声
imread	从图形文件读取图像

[illegible]	[illegible]
imshow	显示图像数据
inv	反矩阵
kron	求右直接乘积
length	向量的长度
linspace	线性间隔所形成的向量
load	读 .mat 文件中的变量
log	自然对数函数
log10	以 10 为底的对数函数
max	求最大值
mean	取平均值
medfilt2	二维中值滤波
min	求最小值
norm	矩阵或向量的范数
ones	元素均为 1 的矩阵
pi	圆周率 π
plot	绘图 (线性)
qtdecomp	四分树分解函数
quad	数值积分
rand	均匀分布的随机数
real	取复变量的实部
reshape	改变矩阵大小
sin	正弦函数
size	求矩阵的大小
sqrt	求平方根
std	求标准差
subplot	分割绘图窗口
sum	求和
svd	进行 SVD 变换
text	加入文字注解
title	加入图形的标题文字
uint8	变换成 unit8 的格式
wcodemat	重新调整数值的范围
whos	列出目前使用中的变量
xlabel	x 轴的坐标
ylabel	y 轴的坐标
zeros	零矩阵

第二章 实验——数学基础

L2.1 二维卷积

原 理

对于一个 LSI 系统，输入序列 $x(n_1,n_2)$ 与脉冲响应 $h(n_1,n_2)$ 的卷积表示成

$$\begin{aligned} y(n_1,n_2) &= x(n_1,n_2) * h(n_1,n_2) \\ &= \sum_{k_1=-\infty}^{\infty}\sum_{k_2=-\infty}^{\infty} x(k_1,k_2)h(n_1-k_1,n_2-k_2) \end{aligned} \tag{L2.1-1}$$

这是一个很重要的数字信号处理运算。以数字图像处理的观点来看，x 若视为数字图像，h 为处理方式，则 y 就是数字图像处理的结果了。

程序范例

1. 目的

给一输入信号及脉冲响应，验证卷积的结果。

2. 程序及说明

```
% 程序 L2_1.m:     二维卷积 %
x=[2 1;3 2];       % 输入信号
h=[-1 1;2 1];      % 脉冲响应
y=conv2(x,h);       %做二维卷积
y
```

3. 结果展示

$$y=\begin{bmatrix} -2 & 1 & 1 \\ 1 & 5 & 3 \\ 6 & 7 & 2 \end{bmatrix}$$

讨 论

用Matlab作卷积可以很快得到我们所要的答案，这正是Matlab为何能够逐渐取代C及Fortran

等程序语言的原因，除了 conv2 这个函数外，Matlab 还有其他许多强大且多样的各种函数可供使用。

动手做

1. 试改变输入讯号及脉冲函数，再做一次。
2. x、h 与 y 三个矩阵的大小之间有何关系？

L2.2 矩阵的直接乘积

原 理

一个 $M\times N$ 大小的矩阵 $\boldsymbol{A}$ 与 $P\times Q$ 大小的矩阵 $\boldsymbol{B}$ 所形成的右直接乘积 (direct product)为一大小为 $PM\times QN$ 且定义如下的矩阵

$$\boldsymbol{C}=\boldsymbol{A}\otimes\boldsymbol{B}=\{a(m,n)\boldsymbol{B}\}=\begin{bmatrix} a(1,1)\boldsymbol{B} & \cdots & a(1,N)\boldsymbol{B} \\ a(2,1)\boldsymbol{B} & \cdots & a(2,N)\boldsymbol{B} \\ \vdots & \vdots & \vdots \\ a(M,1)\boldsymbol{B} & \cdots & a(M,N)\boldsymbol{B} \end{bmatrix} \tag{L2.2-1}$$

同理，左直接乘积则可定义成 $\boldsymbol{C}=\boldsymbol{B}\otimes\boldsymbol{A}=\{b(p,q)\boldsymbol{A}\}$。

程序范例

1. 目的

给二个简单的矩阵 $\boldsymbol{A}$ 与 $\boldsymbol{B}$，验证 $(\boldsymbol{A}\otimes\boldsymbol{B})^{\mathrm{T}}=\boldsymbol{A}^{\mathrm{T}}\otimes\boldsymbol{B}^{\mathrm{T}}$ 与 $(\boldsymbol{A}\otimes\boldsymbol{B})^{-1}=\boldsymbol{A}^{-1}\otimes\boldsymbol{B}^{-1}$ 两个性质。

2. 程序及说明

```
% 程序 L2_2.m: 矩阵的直积%
A=[1 1;-1 1];
B=[1 2;3 4];
A1=A';    B1=B';                % 将矩阵转置
A2=inv(A);    B2=inv(B);        % 求反矩阵
kron1=kron(A,B);                % 求右直接乘积
kron2=kron(A1,B1);
kron3=kron(A2,B2);
kron_left=kron1';
kron_right=kron2;
```

```
invert_left=inv(kron1);
invert_right=kron3;
kron_left
kron_right
invert_left
invert_right
```

3. 结果展示

$$\text{kron_left}=\begin{bmatrix}1 & 3 & -1 & -3\\ 2 & 4 & -2 & -4\\ 1 & 3 & 1 & 3\\ 2 & 4 & 2 & 4\end{bmatrix}\quad \text{kron_right}=\begin{bmatrix}1 & 3 & -1 & -3\\ 2 & 4 & -2 & -4\\ 1 & 3 & 1 & 3\\ 2 & 4 & 2 & 4\end{bmatrix}$$

$$\text{invert_left}=\begin{bmatrix}-1.0000 & 0.5000 & 1.0000 & -0.5000\\ 0.7500 & -0.2500 & -0.7500 & 0.2500\\ -1.0000 & 0.5000 & -1.0000 & 0.5000\\ 0.7500 & -0.2500 & 0.7500 & -0.2500\end{bmatrix}$$

$$\text{invert_right}=\begin{bmatrix}-1.0000 & 0.5000 & 1.0000 & -0.5000\\ 0.7500 & -0.2500 & -0.7500 & 0.2500\\ -1.0000 & 0.5000 & -1.0000 & 0.5000\\ 0.7500 & -0.2500 & 0.7500 & -0.2500\end{bmatrix}$$

讨 论

验证结果完全正确。若矩阵很大，则目视判断验证结果较困难，此时应将左右矩阵相减，再以简易 Matlab 指令判读结果是否为零矩阵，若是则验证正确，否则就是不正确。

动手做

1．试验证以下直接乘积的性质：

(1) $(\boldsymbol{A}+\boldsymbol{B})\otimes\boldsymbol{C}=\boldsymbol{A}\otimes\boldsymbol{C}+\boldsymbol{B}\otimes\boldsymbol{C}$

(2) $(\boldsymbol{A}\otimes\boldsymbol{B})\otimes\boldsymbol{C}=\boldsymbol{A}\otimes(\boldsymbol{B}\otimes\boldsymbol{C})$

(3) $\alpha(\boldsymbol{A}\otimes\boldsymbol{B})=(\alpha\boldsymbol{A})\otimes\boldsymbol{B}=\boldsymbol{A}\otimes(\alpha\boldsymbol{B})$，其中 α 为纯量。

(4) $(\boldsymbol{A}\otimes\boldsymbol{B})(\boldsymbol{C}\otimes\boldsymbol{D})=(\boldsymbol{AC})\otimes(\boldsymbol{BD})$

(5) $\mathrm{tr}(\boldsymbol{A}\otimes\boldsymbol{B})=\mathrm{tr}(\boldsymbol{A})\mathrm{tr}(\boldsymbol{B})$

(6) $\mathrm{rank}(\boldsymbol{A}\otimes\boldsymbol{B})=\mathrm{rank}(\boldsymbol{A})\mathrm{rank}(\boldsymbol{B})$

(7) 若 $\boldsymbol{A}$ 与 $\boldsymbol{B}$ 为么正，则 $\boldsymbol{A}\otimes\boldsymbol{B}$ 亦是。

2. 试写出一左直接乘积 $\boldsymbol{C}=\boldsymbol{B}\otimes\boldsymbol{A}=\{b(p,q)\boldsymbol{A}\}$ 的程序。
3. 试设计一个程序的实验，验证计算 $(\boldsymbol{AC})\otimes(\boldsymbol{BD})$ 确实比计算 $(\boldsymbol{A}\otimes\boldsymbol{B})(\boldsymbol{C}\otimes\boldsymbol{D})$ 来得快。

L2.3　马尔可夫链的转移概率

原　理

马尔可夫链的 k 阶转移概率定义为

$$P_{ji}^{(k)}=P_r[x(n)=S_j\mid x(n-k)=S_i] \tag{L2.3-1}$$

代表原来在状态 S_i 的情况下，经过 k 次状态变迁会成为状态 S_j 的概率。我们可证明 k 阶转移概率与其较低阶的转移概率间有如下的关系

$$P_{ji}^{(k)}=\sum_{q=1}^{Q}P_{jq}^{(l)}P_{qi}^{(k-l)},\quad 0<l<k \tag{L2.3-2}$$

其中 Q 为状态总数。

程序范例

1. 目的

给一初始状态概率，求任意正整数 k 的 k 阶转移概率。

2. 程序及说明

```
% 程序 L2_3.m: 马尔可夫链的转移概率 %
k=8;l=1;
P(1:2,1:2,1)=[0.75 0.25;0.50 0.50];        % 设定概率矩阵（即初始状态概率）
Q=size(P);Q=Q(2);
for kk=2:k
   for i=1:Q
      for j=1:Q
         P(j,i,kk)=0;
         for q=1:Q
            P(j,i,kk)=P(j,i,kk)+P(j,q,l)*P(q,i,kk-l);
         end
      end
   end
end
```

3. 结果展示

k	1	2	3	4	5	6	7	8
$P_{1\|1}^{(k)}$	0.750	0.688	0.672	0.668	0.667	0.667	0.667	0.667
$P_{2\|1}^{(k)}$	0.250	0.312	0.328	0.332	0.333	0.333	0.333	0.333
$P_{1\|2}^{(k)}$	0.500	0.625	0.656	0.664	0.666	0.667	0.667	0.667
$P_{2\|2}^{(k)}$	0.500	0.375	0.344	0.336	0.334	0.333	0.333	0.333

讨 论

以上结果与课文表 2.6-1 的内容相符。用较大的 k 值可明显发现其高阶转移概率的确都会收敛至一个定值

$$P_{\text{limit}}=\begin{bmatrix}0.667 & 0.333\\0.667 & 0.333\end{bmatrix}$$

动手做

1. 对课文内容第二章习题 17 的状况，求其 P_{limit}。

第三章　实验——量化

L3.1　纯量量化器的设计

原　理

最简单的纯量量化方法就是把整个样本值的取值范围均匀地分成 L 个子区间，此乃均匀量化的观念，若子区间大小不一，则得到非均匀量化器。前者通常用于样本值大致均匀落在整个取值范围时，后者则对样本值在取值范围出现机会不均等时有较佳表现，而所谓较佳表现是指在同等位数下，非均匀量化较均匀量化实际引入的噪声小。本实验有关非均匀量化器的设计是根据本书 3.2-2 节所列的步骤来设计的。

程序范例

1. 目的

设图像灰度的概率密度分布为零平均且方差数为 1 的高斯分布

$$p_f(f)=\left(\frac{1}{\sqrt{2\pi}}\right)\exp\left\{-\frac{f^2}{2}\right\} \tag{L3.1-1}$$

由于对称的关系，只考虑正的部分即可，因此设 $d_0=0$，$d_L=\infty$，求最佳量化的判决层 $d_1,d_2,\cdots,d_{L-1}$ 及重构层 $r_1,r_2\cdots,r_L$，其中 L 为偶数。

2. 程序及说明

```
% 程序 L3_1.m: 纯量量化器的设计 %
[r,d]=quantize(2);                  % quantize 函数在本范例中
                                     [r1,d1]=quantize(4);

% 函数 quantize.m:高斯概率密度函数的非均匀量化 %
function [r,d]=quantize(L)                    % L:重构层数
r=linspace(0,0,L); d=linspace(0,0,L+1);
d(1)=0; d(L+1)=500;                           % 设定 d0=0，dL =500 (代表无穷大)
d_inc=0.005; r_inc=0.003; error_bound1=0.002; error_bound2=0.003;
r_estimate=500;tolerance=[1e-4 1e-4];
r_init=0.01;                                   % r(1)值
```

```
check=0;

while check==0
  r(1)=r_init;

  for k=1:L-1
    d(k+1)=d(k);                          % 猜测的 d(k+1) 以 d(k) 开始运算

    % 比对所得的 r_estimate 和 r(k)之差，此循环在 r(k)给定后找出 d(k+1) %
    while abs(r_estimate-r(k))>error_bound1
      d(k+1)=d(k+1)+d_inc;               % 验算不符合略为增加 d_inc
      Int_N=quad('gau1',d(k),d(k+1),tolerance);
      Int_D=quad('gau',d(k),d(k+1),tolerance);
      r_estimate=Int_N/Int_D;
    end
    r(k+1)=2*d(k+1)-r(k);                 % 从 d(k+1) 与 r(k) 求 r(k+1)
  end

  % 从公式算出的 rL_estimate 和 r(k+1) 做比较 %
  IntL_N=quad('gau1',d(L),d(L+1),tolerance);
  IntL_D=quad('gau',d(L),d(L+1),tolerance);
  rL_estimate=IntL_N/IntL_D;
  diff=rL_estimate-r(L);
  relative_error=abs(diff)/rL_estimate;

  if relative_error<error_bound2
    check=1;
  else
    check=0;
    r_init=r_init+r_inc;                  % 不符合条件则重新选取 r(1)
  end
end
% 函数 gau.m:ML 量化器设计中分母的计算式 %
function y=gau(x)
                    y=1/sqrt(2*pi)*exp(-x.^2/2);

% 函数 gau1.m:ML 量化器设计中分子的计算式 %
```

```
function y=gau1(x)
y=x./sqrt(2*pi).*exp(-x.^2/2);
```

3. 结果展示

	$L=2$		$L=4$	
k	d_k	r_k	d_k	r_k
1	0.98	0.454 0	0.51	0.251 0
2		1.506 0	1.065	0.769 0
3			1.765 0	1.361 0
4				2.169 0

讨　论

由于高斯分布的值在零附近较大，导致其量化的值在零附近的差距较小，反之在两旁所差的值较大，也证明非均匀量化对样本值在取值范围出现机会不均等时有较佳的表示。本实验要获得较精准的值必须选取较小的参数，所花费的设计时间也非常可观。

动手做

1. 试求不同的重构层数的结果，例如 $L=6$。
2. 改用不同的概率分布试试看，例如拉普拉氏分布。

L3.2　量化造成的假轮廓

原　理

为了便于计算机的储存，一般常把取值范围分成 2^B 个区间，例如 $B=6,7,\cdots,12$ 、分别把像素的灰度值分成 64,128,…,4 096 个区间。当区间不够时，在图像的灰度值上变化较缓慢的地方会出现假轮廓，这是由于量化过于粗糙，使量化噪声过大所致。反之，若区间数过多又会使计算机处理量大增，所以应适当选取 B 的值。

程序范例

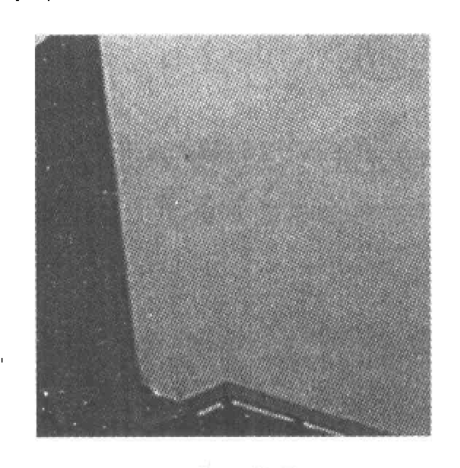
1. 目的

给一张 256 个灰度的图像，将它取较少的位数，探讨其图像在视觉上所产生的变化。

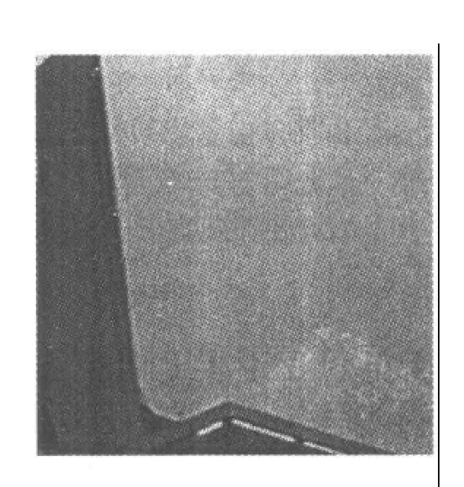
2. 程序及说明

```
% 程序 L3_2.m: 量化造成的假轮廓 %
[X,map]=bmpread('L3_2.bmp');
imshow(X/255,2^8)            %B=8
figure,imshow(X/255,2^6)     %B=6
figure,imshow(X/255,2^4)     %B=4
figure,imshow(X/255,2^3)     %B=3
```

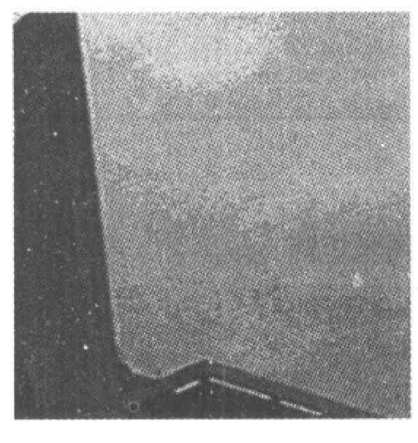
3. 结果展示



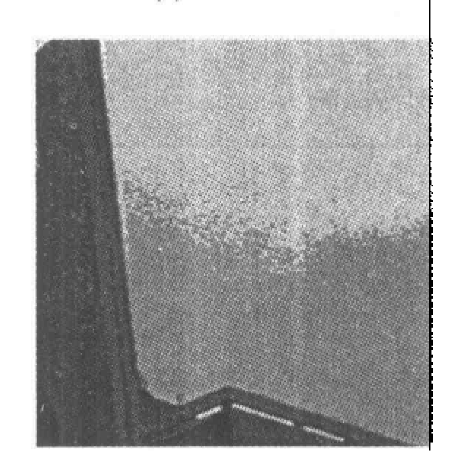
(a) 原图像，位数 $B=8$



(b) $B=6$



(c) $B=4$



(d) $B=3$

图 L3.2-1

讨 论

从图 L3.2-1 可以看出当表示图像的位数愈少时，图像受到的量化噪声较大，故出现假轮廓的机会也愈大。在本例中位数为 4 时，其假轮廓已存在。一般而言，位数为 8 或 256 个灰度时，无法看出假轮廓。

动手做

1. 多用几张图像，试决定平均而言，当 B 为何值时，开始发现有假轮廓。

L3.3　向量量化器码本的产生

原　理

基本上向量量化为一个多对一的映射过程，希望可以利用少数几个向量来代表数字图像中大部分的向量，我们将这些少数的向量分别给予不同的指标，最后将一幅数字图像中所有的向量均只用这些指标来表示。这些少数的向量称为码向量，而其集合即所谓的码本，一般而言码向量是经由许多向量依某种方法反复训练而得。

在第三章中所介绍的 LBG 算法即是用来产生码本的方法。它是利用聚类的原理，依照图像中向量之间的欧几里得距离的大小将其分类，最后找出每一个类别的重心，这些重心就是码向量。

程序范例

1. 目的

将一幅图像利用 LBG 法分别训练出不同大小及维度的码本：cb_64x4、cb_64x16 及 cb_128x4，并评估其效能。

2. 程序及说明

```
% 程序L3_3.m: 向量量化器的码本的产生 %
% 利用LBG法产生三个不同大小与维度的码本 %
load lena                    % 读入一幅 512×512 的图像矩阵
th=0.1;                      % 设定LBG程序停止训练的临界值
nc=64;nd=4;                  % 设定码本大小为 64，维度为 4
[cb_64x4,distortion_64x4]=LBG(X,nc,nd,th);   % LBG函数在本程序范例内
nc=64;nd=16;
[cb_64x16,distortion_64x16]=LBG(X,nc,nd,th);
nc=128;nd=16;
[cb_128x16,distortion_128x16]=LBG(X,nc,nd,th);

% 函数 LBG.m: LBG 训练法 %
function [I_codebook,O_distortion]=LBG(t_image,nc,nd,th)
% I_codebook:最后所得的码本 %
```

```
% O_distortion:向量到最后所得码本之间的平均误差%
% t_image:训练数据%

% Rearrangement of the training image %
[row,col]=size(t_image);
nt=row*col/nd;
R_image=[];
sqrt_nd=sqrt(nd);
x_count=fix(col/sqrt_nd);
y_count=fix(row/sqrt_nd);

for i=1:1:y_count
    for j=1:1:x_count
        r_image=t_image((i-1)*sqrt_nd+1:i*sqrt_nd,(j-1)*sqrt_nd+1:j*sqrt_nd);
        R_image=[R_image;r_image(1:nd)];
    end
end

% Read training data and group the data into training vectors%
ratio=fix(y_count*x_count/nt);
T_image=[];

for i=1:ratio:ratio*nt
    T_image=[T_image;R_image(i,1:nd)];
end
% 制造初始码本 %
ratio=fix(y_count*x_count/nc);
I_codebook=[];

for i=1:ratio:ratio*nc
    I_codebook=[I_codebook;R_image(i,1:nd)];
End

% LBG Algorithm%
disp('Begin LBG algorithm,and wait.......');
Dt=[];
converge=1;
```

```
I_count=1;
Dt(1)=inf;

while(converge)
    % 步骤 1：将码本内容当做重心，将所有的向量分类至这些区间 %
    T_dis=0;

    for i=1:1:nt
        P_dis=inf;
        P_index=0;

        %将每一个向量均归类到与其距离最短的重心那一类 %
        for j=1:1:nc
            dis=((T_image(i,:)-I_codebook(j,:))*(T_image(i,:)-I_codebook(j,:))');
            if dis<P_dis
                P_dis=dis;
                P_index=j; % 归类
            end
        end

        T_dis=T_dis+P_dis/nd;
        id(i)=P_index;
    end

    Dt=[Dt,T_dis/nt];

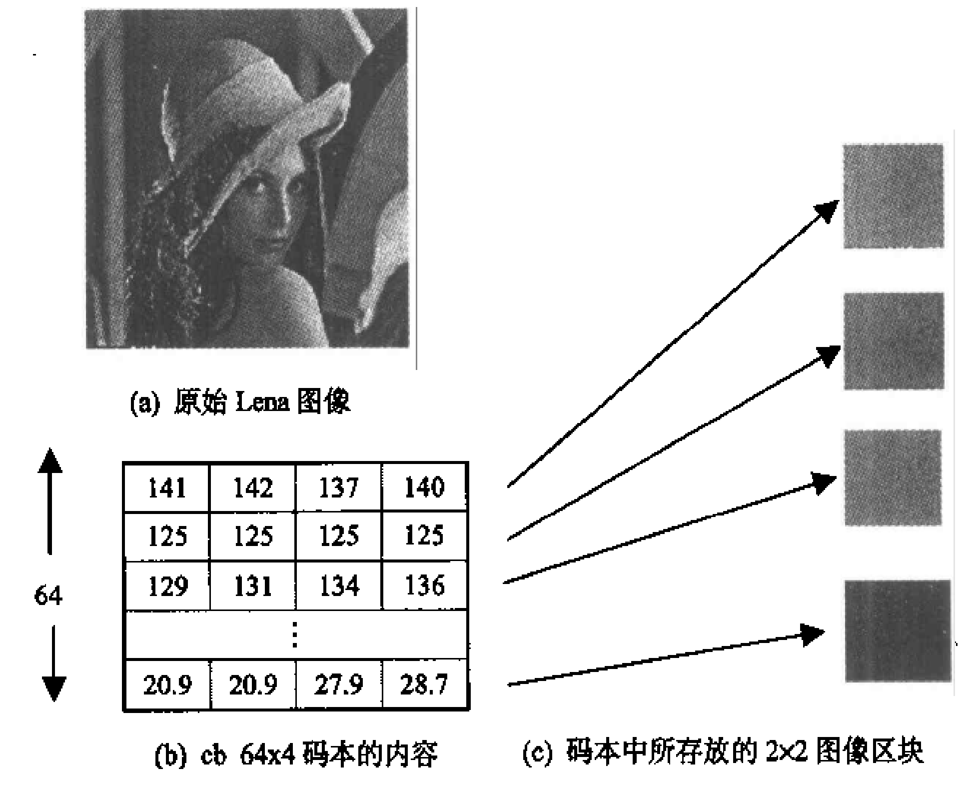
    % 步骤 2：重新计算每一个分类区间的重心并取代先前的码本内容 %
    for i=1:1:nc
        count=0;
        U_codebook=zeros(1,nd);
        for j=1:1:nt
            if id(j)==i
                U_codebook=U_codebook+T_image(j,:);
                count=count+1;
            end
        end
        if count>0
```

```
            I_codebook(i,1:nd)=U_codebook/count;
        end
    end

% 步骤 3 :计算并判断向量到其分类重心之间的平均误差是否小于收敛临界值 th %
    disp('Distortion of current iteration =');
    disp(T_dis/nt);
    I_count=I_count+1;
    % 判断前一次和这一次的平均误差的修正量是否小于 th %
    if abs((Dt(I_count-1)-Dt(I_count))/Dt(I_count-1))<th
        converge=0;
    end
end

O_distortion= T_dis/nt;            % 最后的平均误差
disp('End of the LBG algorithm');
```

3. 结果展示



141	142	137	140
125	125	125	125
129	131	134	136
⋮			
20.9	20.9	27.9	28.7

(a) 原始 Lena 图像

(b) cb 64x4 码本的内容　　(c) 码本中所存放的 2x2 图像区块

图 L3.3-1　对原始 Lena 图像，利用 LBG 算法所求出的 cb_64x4 码本的部分内容

讨　论

由 LBG 算法的实验中发现，码本大小越大，代表分出的向量空间较多，因而每个向量到其所属分类重心的平均距离较小，使得此向量量化器在失真表现上有较佳的结果。但所必须付出的代价则是较长的训练时间，亦即其收敛速度较慢。基本上，LBG 算法的计算复杂度与训练向量数、向量的维度、码本的大小、训练循环次数的乘积成正比，而其中训练循环次数与收敛临界值 *th* 有密切的关系，*th* 愈小则训练循环次数愈多。

在收敛临界值均为 0.1 的状况下，对 cb_64x4、cb_64x16 及 cb_128x4，每一个向量到其所属分类重心 (即码向量) 的平均距离分别为 6.27、9.48 及 5.36，显示码本变小与维度增大通常会导致较大的平均失真。

动手做

1. 试着将训练码本的大小加大成为 256、或是将维度降低为 1，观察其平均失真，亦即每一个码字到其所属分类重心的平均距离。
2. 试着将收敛临界值放大，观察其执行程序的时间是否缩短、及平均失真是否变大。

L3.4　向量量化的编解码

原　理

在向量量化编码时，不同的码向量分别给予不同的指标，借助指标而非向量本身的传送，可以降低表达此图像时所需花费的位数，这就是以向量量化做图像压缩减少数据量的原理。

图 L3.4-1 为向量量化的流程图，其中编码端及解码端的码本均相同。每输入一个向量，编码端会到码本中去寻找和此向量有最短距离的码向量，并将代表此一码向量的指标传送至解码端。解码端主要为查表的操作，将此一指标所对应的码向量取出当做输出向量，完成解码的操作。

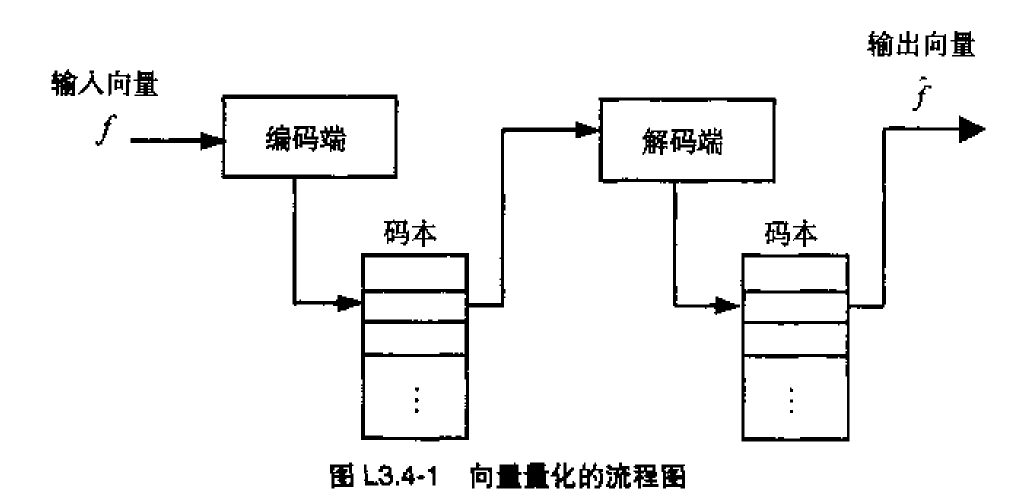


图 L3.4-1　向量量化的流程图

又图像压缩的好坏有两个参数可作为判定的准则，其一为峰值信噪比

$$\mathrm{PSNR} = 20\times\log_{10}\left(\frac{255}{\sqrt{\sum_i \left\|f_i - \hat{f}_i\right\|^2}}\right) \tag{L3.4-1}$$

另一个为压缩率

$$\mathrm{CR} = \frac{\text{原始图像所需位数}}{\text{经压缩图像所需位数}} \tag{L3.4-2}$$

程序范例

1. 目的

利用 L3.3 节的 LBG 程序对一图像训练三个大小及维度均不同的码本:cb_64x4、cb_64x16 及 cb_128x16，再对此图像进行向量量化，并比较其所得的结果。

2. 程序及说明

```
% 程序 L3_4.m: 利用 LBG 训练三个不同大小与维度的码本并分别进行向量量化 %
load lena
th=0.05;
% 训练大小为 64 维度为 4 的码本 cb_64x4 %
nc=64;nd=4;
[cb_64x4]=LBG(X,nc,nd,th);
% 训练大小为 64 维度为 16 的码本 cb_64x16 %
nc=64;nd=16;
[cb_64x16]=LBG(X,nc,nd,th);
% 训练大小为 128 维度为 4 的码本 cb_128x4 %
nc=128;nd=4;
[cb_128x4]=LBG(X,nc,nd,th);

% 使用 cb_64x4 的码本进行向量量化%
[M,N]=size(X);
[Y_64x4,bit_64x4]=VQ(X,cb_64x4);          % VQ 函数在本程序范例中
% 计算 PSNR 和 CR %
MSE=(sum(sum((X-Y_64x4).^2)))/(M*N);
PSNR_64x4=20*log10(255/sqrt(MSE))
CR_64x4=M*N*8/bit_64x4
```

```
% 使用 cb_64x16 的码本进行向量量化 %
[Y_64x16,bit_64x16]=VQ(X,cb_64x16);
MSE=(sum(sum((X-Y_64x16).^2)))/(M*N);
PSNR_64x16=20*log10(255/sqrt(MSE))
CR_64x16=M*N*8/bit_64x16

% 使用 cb_128x4 的码本进行向量量化 %
[Y_128x16,bit_128x4]=VQ(X,cb_128x4;
MSE=(sum(sum((X-Y_128x4).^2)))/(M*N);
PSNR_128x4=20*log10(255/sqrt(MSE))
CR_128x4=M*N*8/bit_128x4

% 函数 VQ.m: 向量量化 %
function [O_data,bit]=VQ(I_data,I_codebook)
% O_data:经向量量化压缩后输出数据 %
% I_data:输入数据 %
% I_codebook:向量量化码本 %
% bit:向量量化所需总位数 %

[row,col]=size(I_data);
[nc,nd]=size(I_codebook);
bit_index=ceil(log2(nc));
bit=0;
sqrt_nd=sqrt(nd);
y_ratio=row/sqrt_nd;
x_ratio=col/sqrt_nd;

% 将每一个向量均拿来做向量量化%
for i=1:y_ratio
    for j=1:x_ratio

        % 取一个图像区块当做输入测试向量 %
        I_block=I_data((i-1)*sqrt_nd+1:i*sqrt_nd,...
            (j-1)*sqrt_nd+1:j*sqrt_nd);

        %寻找码本中和此输入向量最接近的码字 %
```

```
        distortion=sum((((ones(nc,1)*I_block(:)')-I_codebook).^2)')';
        min_distortion=min(find(distortion= =min(distortion)));

        % 将此码字直接取代输入向量,当做输出向量 %
        R_block=reshape(I_codebook(min_distortion,:),sqrt_nd,sqrt_nd);
        O_data((i-1)*sqrt_nd+1:i*sqrt_nd,...
            (j-1)*sqrt_nd+1:j*sqrt_nd)=R_block;
        bit=bit+bit_index;                    % 累加处理每一个向量的位数
    end
end
```

3. 结果展示



(a) 原始 Lena 图像



(b) 码本=cb_64x4，PSNR=32.8，CR=5.33



(c) 码本=cb_64x16、PSNR=28.9，CR=21.33



(d) 码本=cb_128x4，PSNR=34.5，CR=4.57

图 L3.4-2　用三个码本做向量量化压缩后的图像的比较

讨　论

从图 L3.4-2 的结果可以得知，在 LBG 训练时，当码本越大，其所得的向量量化压缩还原的图像失真越小，但相对的其压缩率就越低，这是由于代表码本中这些向量指标的位数较小码本来得多的原因。

另外，当码本中每一个向量的维度越大，即每一个指标所代表的像素越多，则纪录一幅图像所必须花费的指标数目也就越少，相对的所需花费的总位数也就越少，这就是为什么码本维度越大压缩率越高的原因，但其所必须付出的代价则是较高的图像失真。

动手做

1. 试着将训练码本的大小加大成为 256，或是将维度降低为 1，并针对此图像进行向量量化，观察其 PSNR 及 CR。

2. 请随意取一幅异于本实验所采用的图像，并利用本实验所得的码本进行向量量化，另外重新利用此幅图像训练码本，也进行向量量化，观察其 PSNR 及 CR 的差异。

第四章　实验——变换法

L4.1　SVD 变换

原　理

一个 $N \times N$ 的矩阵 $\boldsymbol{A}$ 可以表示成

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^{\mathrm{T}} \tag{L4.1-1}$$

其中 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 的规范正交向量分别是 $\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}$ 与 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ 的特征向量，$\boldsymbol{\Lambda}$ 为一对角矩阵，其主对角线上的原素是 $\boldsymbol{A}$ 的特征值，并可推得

$$\boldsymbol{\Lambda} = \boldsymbol{U}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V} \tag{L4.1-2}$$

此变换法即称为 SVD 变换法。

而因 $\boldsymbol{\Lambda}$ 为一对角矩阵，且主对角线上的元素是 $\boldsymbol{A}$ 的特征值，这些特征值中，有一部分小到可以忽略，因此可借助忽略较小的特征值得到 $\boldsymbol{A}$ 的近似表示。

程序范例

1. 目的

(1) 验证第四章课文内容中有关 SVD 变换的实例。

(2) 对实际图像进行 SVD 变换。

2. 程序说明

```
% 程序L4_1.m: SVD变换 %
A=[ 1 2 1;2 3 2;1 2 1 ];          % 宣告一个矩阵A
[u d v]=svd(A);                   % 对A矩阵进行SVD变换
[nx,ny]=size(A);                  % 算出A矩阵的大小
I=eye(nx,nx);                     % 取一个单位矩阵，其大小为 nx×nx
for i=nx:-1:2
  I(i,i)=0;                       % 将对应到的主对角元素设为 0
end
B=u*I*d*v';                       % 忽略第二和第三个特征值所还原的矩阵
B                                 % 将还原后的矩阵显示出来
```

```
% 对实际图像进行 SVD 变换 %
[X,map]=bmpread('L4_1.bmp');      % 读取图像
A=X(120:220,120:220);             % 将图像的一部分取出，并命名为 A
[nx,ny]=size(A);                  % 算出 A 矩阵的大小
I=eye(nx,nx);                     % 取一个和 A 同样大小的单位矩阵
for i=nx:-1:21                    % 将单位矩阵 I 的部分主对角线元素设为 0
  I(i,i)=0;                       % 以便和 A 的特征值矩阵相乘后
end                               % 消去一部分的特征值
[u d v]=svd(A);                   % 对 A 矩阵进行 SVD 变换
B=u*I*d*v';                       % 将 A 矩阵还原回来并设为 B
subplot(211)
imshow(A,map)                     % 将原始的图像显示出来
subplot(212)
imshow(B,map)                     % 将近似图像显示出来
```

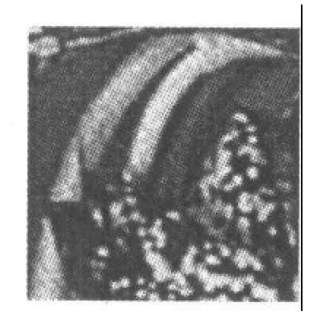
3. 结果展示

B =

1.109 3	1.870 4	1.109 3
1.870 4	3.153 7	1.870 4
1.109 3	1.870 4	1.109 3



(a) 原始图像



(b) 近似的图像

图 L4.1-1

讨 论

由实验的结果显示：***B*** 矩阵和 ***A*** 矩阵两者的差距其实很小，其原因乃是因所舍弃的特征值的 norm 与其他特征值相比起来非常小。再者以实际的图像做范例，由图 L4.1-1 中可看出，在 101 个特征值中只保留 20 个，所还原的图像仍与原始图像所差无几!

动手做

1. 仿程序范例，建立一个 5×5 的矩阵，对其执行 SVD 变换，每次舍弃一个特征值再将

其还原，比较所忽略的特征值对重建矩阵的重要性。

2. 取一个 100×100 的图像，对其执行 SVD 变换，舍弃若干特征值 (设其为零)，将图像还原重建，试计算图像每个像素的平均误差。最后画出纵轴为平均误差，横轴为舍弃特征值个数的曲线图形。

L4.2 图像变换的能量集中能力

原 理

本实验评估不同变换对图像能量重新分配并集中的能力。给一 $N\times N$ 的图像，对此图像执行 $N\times N$ 的变换后，保留左上角低频部分 $n\times n$ 的系数，其余系数设定为零，再进行 $N\times N$ 的反变换。重建图像与原始图像间的误差失真随着 n 的增加而减少，当 $n=N$ 时，误差降为零。以横轴为 n，纵轴为误差量作图可得一曲线，曲线递减至零愈快速，代表该变换的能量集中性 (集中在少数系数上的性能)愈好。

程序范例

1. 目的

比较 DFT 与第二类 DCT 变换的能量集中的能力。

2. 程序说明

```
% 程序 L4_2.m: 图像变换的能量集中能力 %
image1=imread('L4_2.bmp');              % 读取原图像
N=88;                                   % 处理图像大小
image=image1(1:N,1:N);
error1=energy(image,1,N);               % 1 指 DCT
error2=energy(image,2,N);               % 2 指 DFT
plot(1:N,error1,'-.',1:N,error2)        % 虚线是 DCT，实线是 DFT
xlabel('n(保留 n 平方的变换系数) ')
text(40,80,'DFT')
ylabel('平方误差')
text(10,20,'DCT')

% 函数 energy.m: 能量集中 %
function rec_error=energy(image,type,N)   % type :变换方式(DCT or DFT)
% type = 1: DCT, type=2: DFT %
rec_error=zeros(1,N);
```

```
d_image=double(image)/255;              % 将原图像的格式变换为 double 格式
if type==1                              % 做 DCT 变换
   coef_all=dct2(d_image);
else
   coef_all=fft2(d_image);              % 做 DFT 变换
end
for n=1:N
   coef_part=zeros(N,N);
   coef_part(1:n,1:n)=coef_all(1:n,1:n);
     if type==1
     rec_image= idct2(coef_part);       % 做 DCT 反变换
   else
     rec_image=real(ifft2(coef_part));  % 做 DFT 反变换
   end
   % 求误差量 %
   diff=d_image-rec_image;
   rec_error(n)=sum(sum(diff.^2));      % 存放误差量的数据
end
```

3. 结果展示

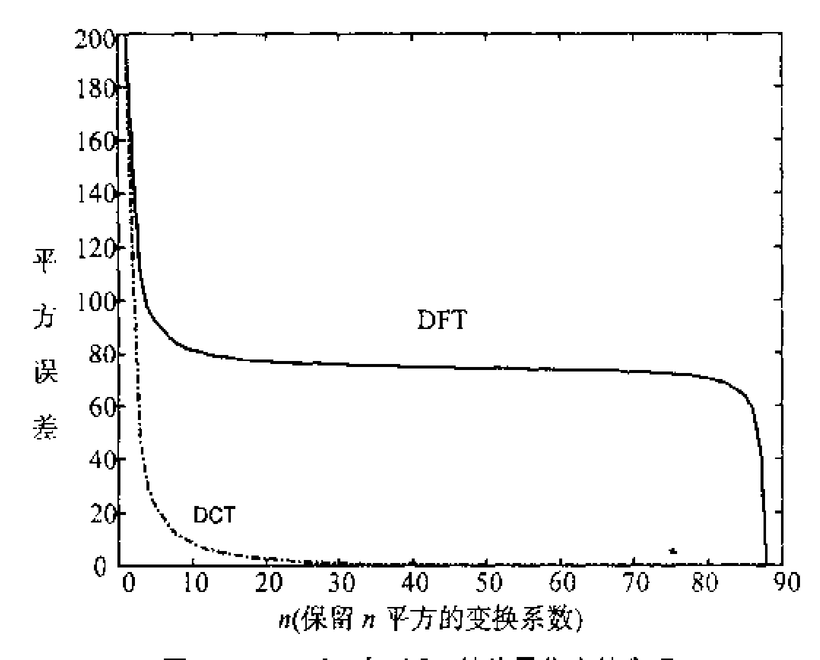


图 L4.2-1　DFT 与 DCT 的能量集中的表现

讨　论

从图 L4.2-1 可知，在 $n<8$ 时两个变换的表现相差不多，愈到后面相差愈多，DCT 的曲

线下降比 DFT 快．由此得知，在处理较大数据时，DCT 比 DFT 好．

动手做

1. 延续上例的做法，但 $n\times n$ 系数从右下角 (高频处) 往左上角 (低频处) 方向取，画出类似图 L4.2-1 的图形，并讨论这两个曲线图形的差异．

2. 加入自选的第三个变换，将对应的第三条曲线加入图 L4.2-1 中，并讨论新的结果．

L4.3　小波变换

原　理

小波变换可将信号的不同频率成分分解出来．给一图像，经过一层的小波变换可得到如图 4.11－10(b) 所示的分解结果，其中 $f_1^1(x,y)$ 表示低频成分，$f_4^1(x,y)$ 则表示最高频成分．另外若连续做 p 次小波变换可将所有 $f_2^p(x,y)$ 看成小波函数在水平方向的影响，$f_3^p(x,y)$ 看成在垂直方向的影响，$f_4^p(x,y)$ 看成在对角方向上的影响．

程序范例

1. 目的

对图像做小波变换，并从视觉上感受各频带系数所代表的意义．下面做了两个实验：(a) 部分说明小波变换的阶层结构，(b) 部分说明小波变换各频带系数的影响．在(b)中我们对图像做四层的小波变换后分别将所有 $f_2^p(x,y)$ 、$f_3^p(x,y)$ 、$f_4^p(x,y)$ 的系数舍弃，其结果将反应到图像的各方向上．

2. 程序及说明

```
% 程序 L4_3a.m: 小波变换的多重解析结构 %
% dwtper2(): 为 Matlab 图像工具箱中做离散小波变换的函数 %
%            其中'db1'表示使用滤波器长度为 1 的 daubechies 小波函数 %
% idwtper2(): 重建时使用的反变换函数 %
% wcodemat(): 经小波变换后必须将系数重新编码使其落在 0~255 之间，以方便显示．
% 以上均为 Wavelet 工具箱中所提供的函数%


clear;
load woman                                   % 载入图像
imshow(X,map), xlabel('(a)')
```

```
% 第一层小波变换 %
nbcol=size(map,1);
[LL1,LH1,HL1,HH1]=dwtper2(X,'db1');     % 小波变换产生各频带系数
coef11=wcodemat(LL1,nbcol);              % 调整频带系数大小
coef12=wcodemat(LH1,nbcol);
coef13=wcodemat(HL1,nbcol);
coef14=wcodemat(HH1,nbcol);

% 第二层小波变换 %
[LL2,LH2,HL2,HH2]=dwtper2(LL1,'db1');  % 对低频系数再做一次小波变换
coef21=wcodemat(LL2,nbcol);              % 调整频带系数大小
coef22=wcodemat(LH2,nbcol);
coef23=wcodemat(HL2,nbcol);
coef24=wcodemat(HH2,nbcol);
figure;
imshow([[coef21,coef22;coef23,coef24],coef12;coef13,coef14],map)
xlabel('(b)')

% 重建 %
rLL1=idwtper2(LL2,LH2,HL2,HH2,'db1');   % 反小波变换
rX=idwtper2(rLL1,LH1,HL1,HH1,'db1');
figure;
imshow(rX,map), xlabel('(c)')

err=sum(sum(abs(X-rX)))

% 程序 L4_3b.m: 小波变换系数与图像的方向性 %
clear;
[X,map]=bmpread('L4_3b.bmp');            % 载入图像
subplot(221), imshow(X,map), xlabel('(a)')

% 小波变换 %
nbcol=size(map,1);
[LL1,LH1,HL1,HH1]=dwtper2(X,'db8');     % 小波变换产生各频带系数
[LL2,LH2,HL2,HH2]=dwtper2(LL1,'db8');   % 滤波器长度为 8
[LL3,LH3,HL3,HH3]=dwtper2(LL2,'db8');
[LL4,LH4,HL4,HH4]=dwtper2(LL3,'db8');
```

```
% 重建(某些频带系数以零取代) %
altzero4=zeros(size(LL4));
altzero3=zeros(size(LL3));
altzero2=zeros(size(LL2));
altzero1=zeros(size(LL1));

% 反小波变换
rLL3=idwtper2( LL4,altzero4,HL4,HH4,'db8');
rLL2=idwtper2(rLL3,altzero3,HL3,HH3,'db8');
rLL1=idwtper2(rLL2,altzero2,HL2,HH2,'db8');
rX=idwtper2(rLL1,altzero1,HL1,HH1,'db8',size(X));
difrtH=rX-X;                       % 计算去掉各层第二频带成分后重建的图
                                   % 与原图的差异
discardH=wcodemat(difrtH,nbcol);  % 观察去掉第二频带成分后的影响
subplot(222), imshow(discardH,map), xlabel('(b)')

rLL3=idwtper2( LL4,LH4,altzero4,HH4,'db8');
rLL2=idwtper2(rLL3,LH3,altzero3,HH3,'db8');
rLL1=idwtper2(rLL2,LH2,altzero2,HH2,'db8');
rX=idwtper2(rLL1,LH1,altzero1,HH1,'db8',size(X));
difrtV=rX-X;                       % 计算去掉各层第三频带成分后重建的图
                                   % 与原图的差异
discardV=wcodemat(difrtV,nbcol);  % 观察去掉第三频带成分后的影响
subplot(223), imshow(discardV,map), xlabel('(c)')

rLL3=idwtper2( LL4,LH4,HL4,altzero4,'db8');
rLL2=idwtper2(rLL3,LH3,HL3,altzero3,'db8');
rLL1=idwtper2(rLL2,LH2,HL2,altzero2,'db8');
rX=idwtper2(rLL1,LH1,HL1,altzero1,'db8',size(X));
difrtX=rX-X;                        % 计算去掉各层第四频带成分后重建的
                                    % 图与原图的差异
discardX=wcodemat(difrtX,nbcol);   % 观察去掉第四频带成分后的影响
subplot(224), imshow(discardX,map), xlabel('(d)')
```

3. 结果展示



(a) 原始图像



(b) 变换后系数图



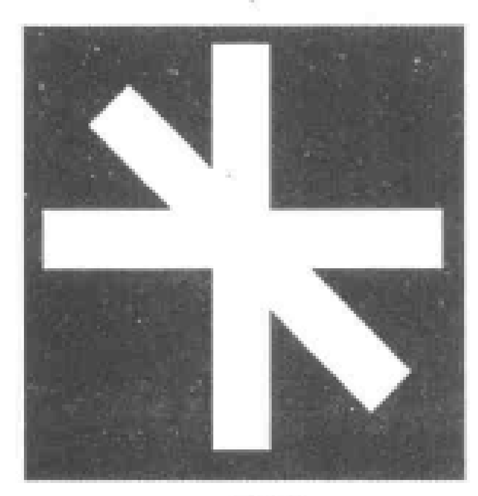
(c) 重建后图像

图　L4.3-1

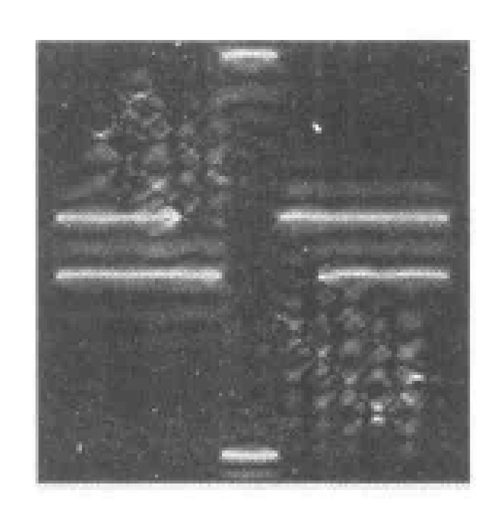
原始图像与重建图像的误差

err =

4.2237e – 009　(小到几乎可以忽略)

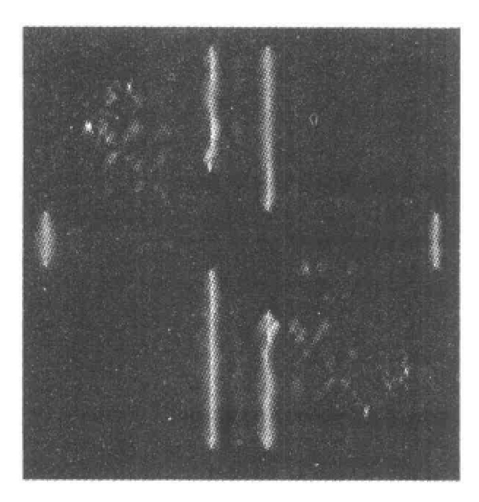


(a) 原图像

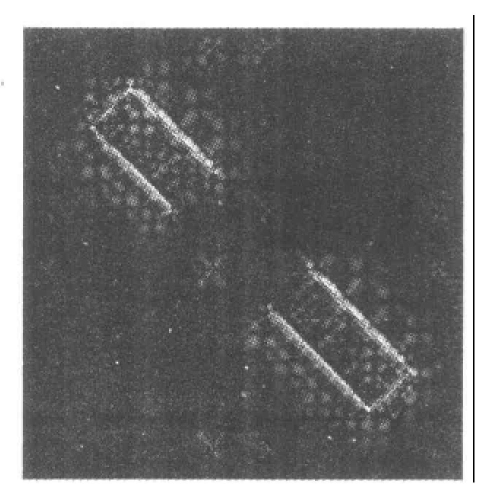


(b) 去掉 $f_2^{\Gamma}(x, y)$

图　L4.3-2 (待续)



(c) 去掉 $f_3^p(x,y)$



(d) 去掉 $f_3^p(x,y)$ 系数后与原图像之差

图　L4.3-2 (续)

讨　论

由 L4.3-1 可看出小波变换多解析及完全重建不失真的特性。仔细观察图 L4.3-1(b)，其噪声成分大都在最高频带 $f_4^1(x,y)$ 内，越往左上则图像轮廓越清楚。从此实验可知，信号的滤波与压缩等应用均可经小波变换轻易的做到。另外由图 L4.3-2 的 (b) 到 (d) 表示舍弃各频带系数重建后与原图像的差值，相差越多则越白。此实验显现小波变换后各频带系数分别对水平、垂直及对角三个方向的视觉效应。

动手做

1. 重做 (a) 部分，但将最高频带系数舍弃，计算其重建后图像与原图像的误差，并观察其对原图像所造成的影响。

第五章 实验——图像增强

L5.1 直方图均衡化法

原 理

灰度图像的直方图就是灰度的一种统计图，从此统计图中我们可以得知图像在亮度上的特性，包括其动态分布的范围。当一张图像的灰度集中在某一段区域内时，图像的对比度会相当差。为改善这种情形，我们使用 5.1-2 节直方图均衡化的方法来处理图像。

直方图均衡化的意思就是将灰度平均分配给图像中的所有像素，使灰度的直方图尽可能呈现均匀分布，如此可提高图像的对比度，图像看起来会比较清晰。

程序范例

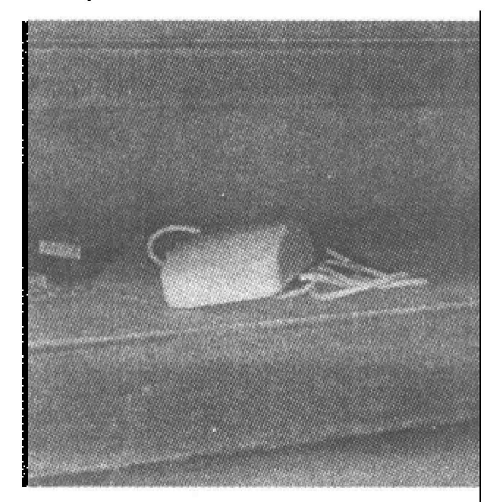
1. 目的

对一张低对比的图像，以直方图均衡化法使其对比度增强，同时比较处理前及后的图像及其直方图。

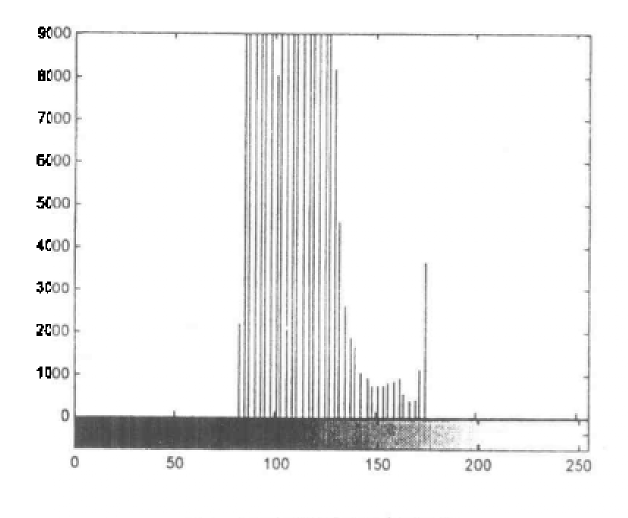
2. 程序及说明

```
% 程序 L5_1.m: 直方图均衡化法%
% image_1  原图像；image_2  执行直方图均衡化后的图像%
image_1=imread('L5_1.bmp');     % 读取图像
image_2=histeq(image_1);        % 对上述图像执行直方图均衡化
imshow(image_1)                 % 显示原图像
figure,imshow(image_2)          % 显示处理后的图像
figure,imhist(image_1)          % 显示原图像的直方图
figure,imhist (image_2)         % 显示处理后图像的直方图
```

3. 结果展示

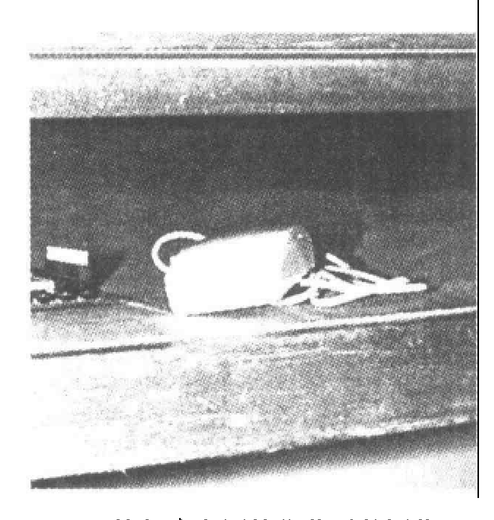


(a) 对比度低的图像

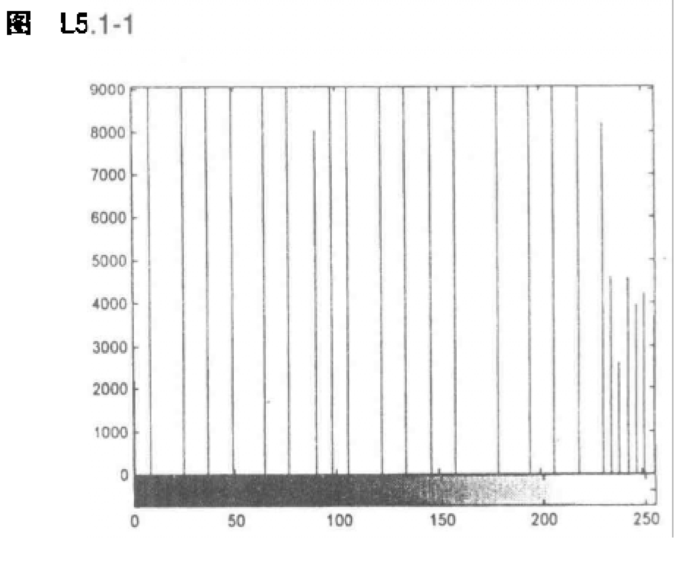


(b) (a)中图像的直方图

图　L5.1-1



(a) 执行直方图均衡化后的图像



(b) (a)中图像的直方图

图　L5.1-2

讨　论

从图 L5.1-1 与 L5.1-2 可以看出，原图像的灰度值非常集中，导致其对比度效果差，而经过直方图均衡化处理后，图像灰度值重新分配，让整幅图像对比度提高许多。

动手做

1. 利用 Matlab 中 Image Toolbox 的 imhist，将原图像的灰度值范围重新定义在 50 到 100、150 及 200，并比较其结果。

2. 利用 Matlab 中 Image Toolbox 的 imhist，将原图像的灰度值范围分为 50 ~ 100 及 150 ~ 200 两个范围，并比较其结果。

L5.2　平滑滤波器

原　理

● **低通空间滤波器**

低通滤波器中常见的均化模板是将模板中所有灰度值加总后求其平均，然后将平均值写入模板中间所对应点的像素。又因为一次只处理一个像素的值，旁边的像素在下一次的模板移动后再作处理，因此又称为移动平均滤波器。此滤波器可去除噪声，但也会产生图像模糊的反效果。

常用的空间模板是由 3×3 共 9 个 1/9 的系数所组成

$$\begin{matrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{matrix} \tag{L5.2-1}$$

也可提高模板中间的加权值变成

$$\begin{matrix} 1/10 & 1/10 & 1/10 \\ 1/10 & 2/10 & 1/10 \\ 1/10 & 1/10 & 1/10 \end{matrix} \tag{L5.2-2}$$

或是提高模板中间与四邻接点的加权值

$$\begin{matrix} 1/16 & 1/8 & 1/16 \\ 1/8 & 1/4 & 1/8 \\ 1/16 & 1/8 & 1/16 \end{matrix} \tag{L5.2-3}$$

注意到以上模板的加权值总和为 1，如此才不至于造成空间卷积运算后图像过亮或过暗的结果。

● **中值滤波**

模板的排序运算是一种非线性的滤波方式，首先对在模板里所有像素的灰度大小做排序，接着选取排序在中间的灰度值，再将此值代入所对应的像素中，如此便完成一个像素的滤波操作，此法不但可以滤掉图像中突起的高频噪声部分，对于图像的边缘，通常也能够给予适当的保持。

程序范例

1．目的

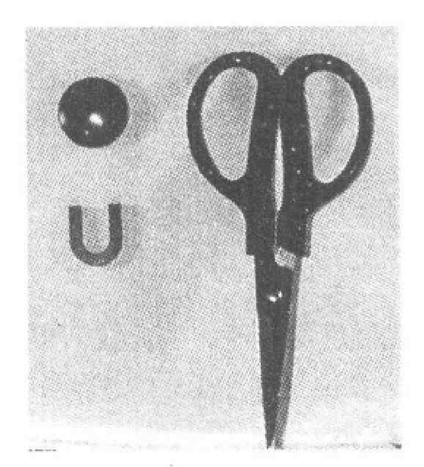
给一张受污染的图像，分别以低通滤波器及中值滤波的方法去处理，比较其处理结果。

2．程序及说明

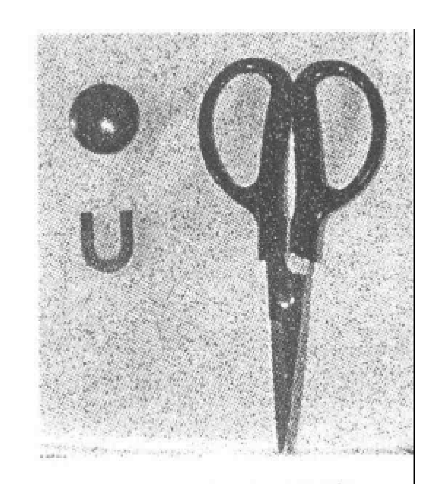
```
% 程序 L5_2.m: 平滑滤波器 %
% Image: 原图像 %
% Image_noisy: 受污染的图像 %
```

```
% Image_low: 执行低通滤波器后的图像 %
% Image_med: 执行中值滤波器后的图像 %
Image=imread('L5_2.bmp');                        % 读取图像
% 将胡椒盐式的噪声加入原图像中，其中 0.02 是噪声密度 %
Image_noisy= imnoise (Image ,'salt & pepper',0.06);
Image2_noisy=double(Image_noisy)/255;            % 变为 double 格式
h=[1/9 1/9 1/9;1/9 1/9 1/9 ; 1/9 1/9 1/9];       % 滤波器的子模板
Image_low=filter2(h, Image2_noisy);              % 执行低通滤波
Image_med=medfilt2(Image_noisy,[3 3]);           % 中值滤波采用 3×3 矩阵
% 显示图像 %
imshow(Image)
figure,imshow(Image_noisy)
figure,imshow(Image_low)
figure,imshow(Image_med)
```

3. 结果展示

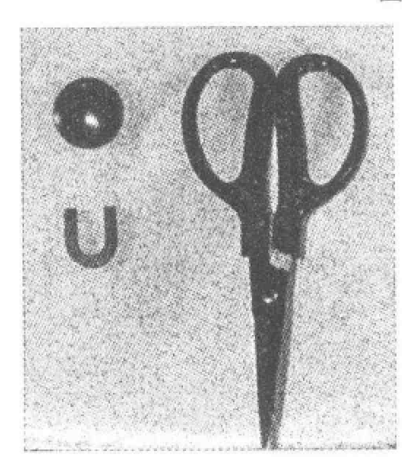


(a) 原图像

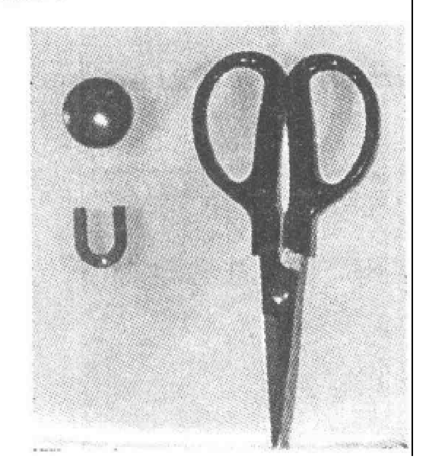


(b) 原图像加入噪声

图 L5.2-1



(a) 低通滤波



(b) 中值滤波

图 L5.2-2　将受污染的图像滤波处理后的结果

讨　论

根据图 L5.2-2 中两张用不同方法处理受污染图像的结果可以发现，在图像的清晰度上，用中值滤波处理比用低通滤波处理效果还要好。如在原理中所述，因为低通滤波是一种平均的方法，其边缘效果被破坏，使图像变模糊，反之中值滤波是一种排序方法，其图像及边缘的保持效果较好。而在噪声处理上，中值滤波处理后的图像内，噪声几乎看不到；而在低通滤波处理后的图像中，其噪声还是存在，只是较模糊不明显而已。总之，在噪声的处理上，中值滤波比低通滤波效果更佳。

动手做

1. 同上例，将模板大小改为 5×5，与上例比较其差异。
2. 将上例中所加的噪声变为高斯噪声，重做本实验，并讨论其结果。

L5.3　同态滤波器

原　理

假设一个图像 f 可由光强度 i 与物体反射光强度 r 的乘积来决定，即

$$f = i \cdot r \tag{L5.3-1}$$

经过同态滤波后其结果会改变图像光强度与反射光强度的特性，因此我们可以做到同时降低图像动态范围，又增加对比度的结果。所用的方法及过程如图 L5.3-1 所示。

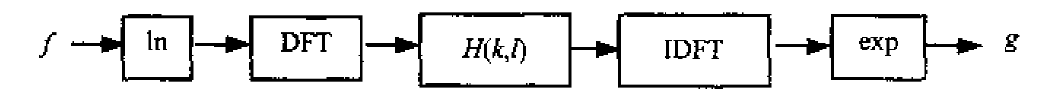


图 L5.3-1　同态滤波的程序

在图 L5.3-1 中，$H(k,l)$ 为一 n 阶的 Butterworth 高通滤波器，其转移函数定义如下

$$H(k,l) = \gamma_L + \frac{\gamma_H}{1+[D_0/D(k,l)]^{2n}} \tag{L5.3-2}$$

上式中 D_0 为截止频率，$D(k,l) = (k^2 + l^2)^{1/2}$，$\gamma_L$ 与 γ_H 为可调整的参数。可针对不同的图像采用不同的参数值，通常选择 $\gamma_L < 1$，$(\gamma_L + \gamma_H) > 1$，会获得所要的结果。

程序范例

1. 目的

采用同态滤波增强图像的对比度，并使图像较暗的地方变得更清楚。

2. 程序及说明

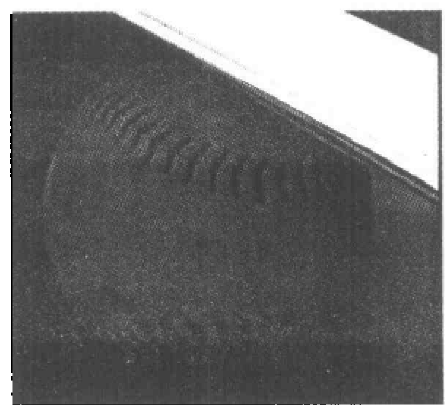
```
% 程序 L5_3.m: 同态滤波器 %
clear
[image_0,map]=bmpread('L5_3.bmp');              % 读取图像
image_1=log(image_0+1);                         % 取自然对数
image_2=fft2(image_1);                          % 做二维傅立叶变换

% 产生高通 Butterworth 滤波器 %
n=3;
D0=0.05*pi;
rh=0.8;
rl=0.5;
[row,col]=size(image_2);
for k=1:1:row
   for l=1:1:col
      D1(k,l)=sqrt((k^2+l^2));
      H(k,l)=rl+(rh/(1+(D0/D1(k,l))^(2*n)));
   end
end

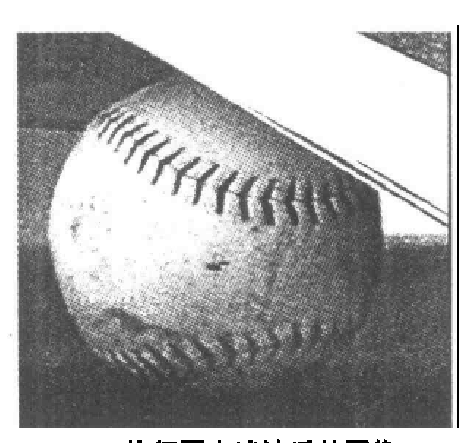
image_3=(image_2.*H);                           % 输入图像通过滤波器
image_4=ifft2(image_3);                         % 做二维反傅立叶变换
image_5=(exp(image_4)-1);                       % 取指数函数

figure(1)
imshow(image_0,map)                             % 显示原图像
figure(2)
imshow(real(image_5),map)                       % 显示处理后的图像
```

3. 结果展示



(a) 原图像



(b) 执行同态滤波后的图像

图　L5.3-2

讨　论

从图 L5.3-2 可以得知，经过同态滤波后，整张图像较之前的图像更清楚且较暗的地方也变的更明亮清晰。若将原图像以直方图均衡化法处理后 (如图 L5.3-3)，再相比较可发现，虽然都是将图像变得更清楚，但是在仔细观察后仍可发现图像经过同态滤波后，对比度比直方图均衡化法的结果更明显。



图 L5.3-3　执行直方图均衡化后的图像

动手做

1. 试着选取不同的阶数 n 、截止频率 D_0 及参数 γ_L 与 γ_H ，看会得到何种结果？

第六章　实验——图像恢复

L6.1　最小平方滤波器

原　理

图像恢复的主要目的是去改善一幅品质遭受恶化的图像。恢复是一种过程，此种过程试图利用对恶化现象发生之前的了解，建立恶化的模型，再运用相反的过程来重建或恢复图像。

最小平方 Wiener 滤波器图像恢复法的公式如 (6.4-11) 式

$$\hat{F}(k,l)=\left[\frac{H^{*}(k,l)}{H(k,l)^{2}+\gamma[S_{\eta}(k,l)/S_{f}(k,l)]}\right]G(k,l) \tag{L6.1-1}$$

式中 $k,l=0,1,2,\cdots,N-1$，$\gamma=1/\alpha$。

如果不知道噪声的统计性质，即 $S_{\eta}(k,l)$ 和 $S_{f}(k,l)$ 未知时，则 (L6.1-1) 式可表示为

$$\hat{F}(k,l)\approx\left[\frac{1}{H(k,l)}\frac{|H(k,l)|^{2}}{|H(k,l)|^{2}+K}\right]G(k,l) \tag{L6.1-2}$$

式中 K 近似为一常数。此式可以使恶化图像得到某种程度的恢复，但未必是最佳恢复。

在实际的应用中，$H(k,l)$ 可以用解析的方法得到。本实验考虑一个因照相机水平晃动而被模糊的图像，并用 Wiener 滤波器来恢复。而长度 $2d$ 的照相机水平移动模糊如 (6.6-7) 式

$$h(m,n)=\begin{cases}0, & n\neq 0,-\infty<m<\infty\\ \dfrac{1}{2d}, & n=0,-d\leqslant m\leqslant d\end{cases} \tag{L6.1-3}$$

由 (L6.1-3) 式再经离散傅立叶变换，可得图像因照相机水平晃动而被模糊的转移函数 $H(k,l)$，再用最小平方 (Wiener) 滤波器恢复法公式 (L6.1-2) 式，调整适当的 K 值将模糊的图像恢复。

程序范例

1. 目的

对一张因照相机水平晃动而被模糊且加有噪声的图像，使用最小平方 (Wiener) 滤波器恢复，同时比较 $K=0$ 及 0.000 1 两种不同参数值时图像恢复的结果。

2. 程序结果及说明

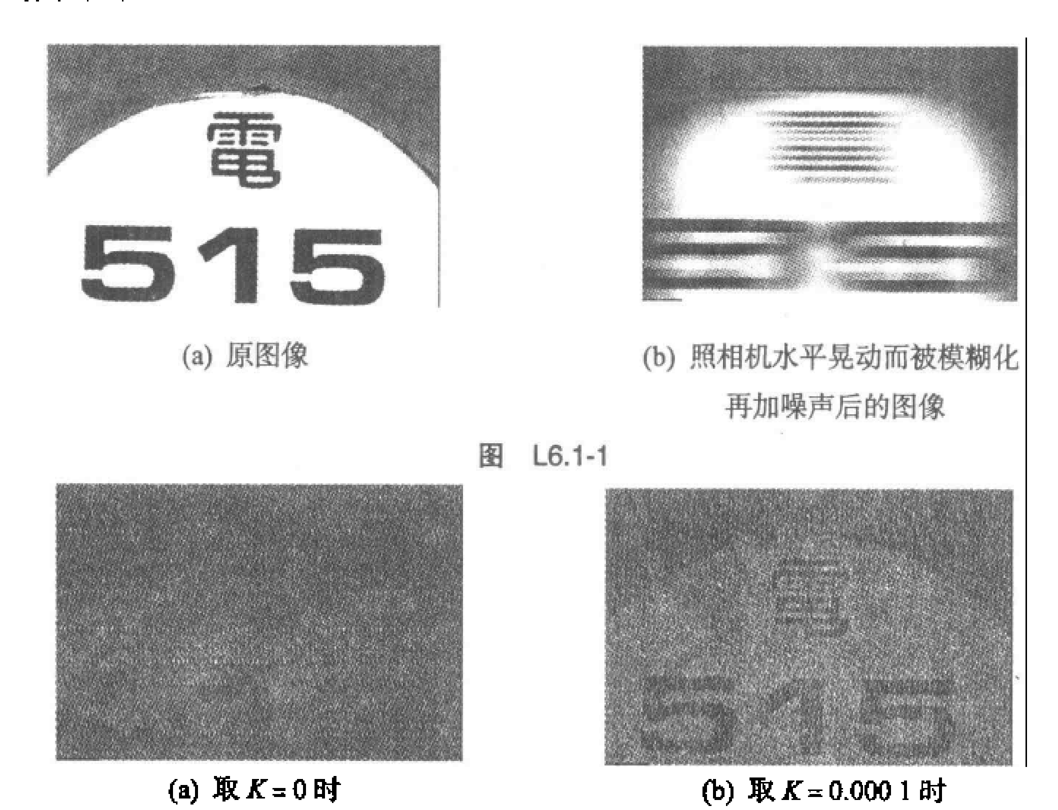
```
% 程序 L6_1.m: 最小平方滤波器 %
% 设定长度 2d 的照相机水平移动模糊的函数 h %
clear                                   % 清除变量
d=50;                                   % 设定长度
h=zeros(2*d+1,2*d+1);
h(d+1,1:2*d+1)=1/(2*d);                 % 函数 h

%将图像模糊且加入噪声%
f=imread('L6_1.bmp');                   % 读取图像
[m,n]=size(f);                          % 图像大小
fe=zeros(m+2*d,n+2*d);
fe(1:m,1:n)=f;                          % 扩增 f
he=zeros(m+2*d,n+2*d);
he(1:2*d+1,1:2*d+1)=h;                  % 扩增 h
F=fft2(fe);
H=fft2(he);
ns=5*rand(m+2*d,n+2*d);                 % 产生噪声
g= ifft2(F.*H)+ns;                      % 被模糊化且加有噪声的图像
G=fft2(g);

% 利用最小平方 (Winer) 滤波器恢复图像 %
K=0;                                    % 设定 K 值
F_estimate=((H.^2)./(H.^2+K)).*G./H;    % 最小平方 (Wiener) 滤波器恢复公式
f_estimate=real(ifft2(F_estimate));     % 恢复后的图像

% 结果展示 %
imshow(f)                               % 显示模糊化前的图像
% 显示模糊再加噪声后的图像 %
figure
imshow(g(d+1:m+d,d+1:n+d),[min(g(:)) max(g(:))])
% 显示恢复后图像 %
figure
imshow(f_estimate(1:m,1:n),[min(f_estimate(:)) max(f_estimate(:))])
```

3. 结果展示



(a) 原图像

(b) 照相机水平晃动而被模糊化再加噪声后的图像

图　L6.1-1

(a) 取 $K=0$ 时

(b) 取 $K=0.000\,1$ 时

图 L6.1-2　恢复的结果

讨　论

从上几张图像及实验的经验发现，对照相机水平晃动而被模糊化的图像加上噪声时，必须调整适当的 K 值才能利用最小平方 (Wiener) 滤波器清楚恢复图像。

动手做

1. 尝试其他图像，重新执行程序，观察恢复结果。
2. 采用不同的噪声大小，调整适当的 K 值恢复图像。

L6.2　迭代盲目去卷积法

原　理

假设我们得到一个受污染的图像 $g(m,n)$ 是由原始图像 $f(m,n)$ 与点扩散函数 (PSF) $h(m,n)$ 的卷积所构成，则我们可以采用迭代盲目去卷积法来恢复原始图像，其方块图如图 6.6-3 所示。

一开始我们假设一初始估测图像 $f_0(m,n)$，接着就在图像与频率域间交替变换运算，

再加入一些有关图像及 PSF 信息的条件限制。在第 r 次迭代时，傅立叶域的条件限制为

$$\widetilde{H}_r(k,l)=\frac{G(k,l)\hat{F}_{r-1}^*(k,l)}{\left|\hat{F}_{r-1}(k,l)\right|^2+\alpha\Big/\left|\widetilde{H}_{r-1}(k,l)\right|^2} \tag{L6.2-1}$$

$$\widetilde{F}_r(k,l)=\frac{G(k,l)\hat{H}_{r-1}^*(k,l)}{\left|\hat{H}_{r-1}(k,l)\right|^2+\alpha\Big/\left|\hat{F}_{r-1}(k,l)\right|^2} \tag{L6.2-2}$$

其中 α 是相加性噪声的能量。

在本实验中，我们不考虑噪声所造成的影响，亦即 $\alpha=0$。则 (L6.6-1) 及 (L6.6-2) 式可分别化简为

$$\widetilde{H}_r(k,l)=\frac{G(k,l)}{\hat{F}_{r-1}(k,l)} \tag{L6.2-3}$$

$$\widetilde{F}_r(k,l)=\frac{G(k,l)}{\hat{H}_{r-1}(k,l)} \tag{L6.2-4}$$

此处有两个主要的问题存在：

1. 因为反函数的运算，使得具有低数值区域的函数在定义上会有困难。
2. 在 $\hat{F}_{r-1}(k,l)$ 或 $\hat{H}_{r-1}(k,l)$ 的零点所在的特别区域的频率上无法获得任何信息。

为了解决这两个问题，此处加入了傅立叶域的条件限制 (Ayers and Dainty [1988])：

若 $|G(k,l)|<\text{noise level}$,

$$\widetilde{F}_{r+1}(k,l)=\hat{F}_r(k,l) \tag{L6.2-5a}$$

若 $\left|\hat{H}_r(k,l)\right|\geqslant|G(k,l)|$,

$$\widetilde{F}_{r+1}(k,l)=(1-u)\hat{F}_r(k,l)+u\frac{G(k,l)}{\hat{H}_r(k,l)} \tag{L6.2-5b}$$

若 $\left|\hat{H}_r(k,l)\right|<|G(k,l)|$,

$$\frac{1}{\widetilde{F}_{r+1}(k,l)}=\frac{1-u}{\hat{F}_r(k,l)}+u\frac{\hat{H}_r(k,l)}{G(k,l)} \tag{L6.2-5c}$$

其中 $0\leqslant u\leqslant 1$。

在模糊的条件限制中，我们将反傅立叶后的图像 $\widetilde{h}_r(m,n)$ 中像素值为负值者设为 0。而在图像域的条件限制中，将所猜测的物体图像的区域范围外的像素值设为 0。

程序范例

1. 目的

将一幅被模糊的图像，利用迭代盲目去卷积法来恢复其原始的图像。

2. 程序及说明

```
% 程序 L6_2.m: 迭代盲目去卷积法 %
clear
% 制作模糊的图像%
[h0,map]=bmpread('L6_2a.bmp');          % 读取图像 h0
[f0,map]=bmpread('L6_2b.bmp');          % 读取图像 f0
[y1,y2]=size(h0);                       % 量测 h0 大小为 y1×y2
[u1,u2]=size(f0);                       % 量测 f0 大小为 u1×u2
y=y1+y2;
u=u1+u2;
he=zeros(y,y);            % 将(y1+y2)×(y1+y2)的零矩阵令为 he
fe=zeros(u,u);            % 将(u1+u2)×(u1+u2)的零矩阵令为 fe

% 将 h0 扩张为(y1+y2)×(y1+y2)的矩阵 %
he([y/4:(y/4)+y1-1],[y/4: (y/4)+y2-1])=h0;
% 将 f0 扩张为(u1+u2)×(u1+u2)的矩阵 %
fe([u/4:(u/4)+u1-1],[u/4:(u/4)+u2-1])=f0;
h1=he;
f1=fe;
F=fft2(f1);
H=fft2(h1);
G=F.*H;                       % 将 f 与 h 做卷积得到模糊图像 g
g1=ifft2(G);
% 显示 PSF 的图像 %
figure,imshow(real(h1),[min(real(h1(:))) max(real(h1(:)))]);
% 显示原始图像 %
figure,imshow(real(f1),[min(real(f1(:))) max(real(f1(:)))]);
% 显示模糊的图像 %
figure,imshow(real(g1),[min(real(g1(:))) max(real(g1(:)))]);

% 开始执行迭代盲目去卷积法 %
f0_=rand(64,64);              % 猜测的初始图像
% figure,imshow(real(f0_),[min(real(f0_(:))) max(real(f0_(:)))]);
F0_=fft2(f0_);
% 加上傅立叶域的条件限制 %
H0=G.*(F0_.^(-1));
```

```
h0=ifft2(H0);
% 加上模糊的条件限制，将猜测图像的区域范围的外设为零 %
h0([1:y/4],:)=0;h0([(y/4)+y1-1:y],:)=0;
h0(:,[1:y/4])=0;h0(:,[(y/4)+y1-1:y])=0;
h0_=h0;
H0_=fft2(h0_);
u=0.9;                          % 设定参数

% 进行循环迭代的步骤 %
for i=1:1500                    % 设定迭代的次数
    i
    for k=1:64
        for l=1:64
            % 加上傅立叶域的条件限制 %
            if abs(G(k,l))<=0.05;       % 设定 noise level 为 0.05
                F1(k,l)=F0_(k,l);
            elseif abs(H0_(k,l))>=abs(G(k,l));
                F1(k,l)=(1- u).*F0_(k,l)+...
                    u.*(G(k,l).*(H0_(k,l).^(-1)));
            else
                F1(k,l)=(((1-u).*(F0_(k,l).^(-1)))+...
                    u.*(H0_(k,l).*(G(k,l).^(-1)))).^(-1);
            end
        end
    end
    f1=ifft2(F1);

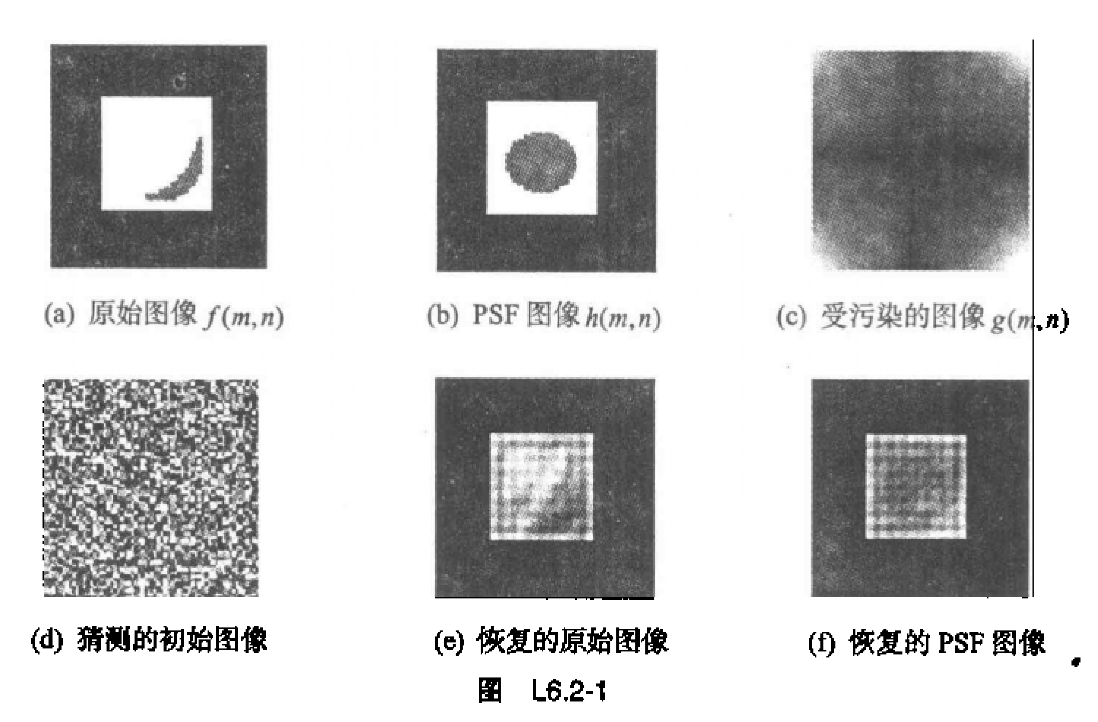
    % 加入图像域的条件限制，将负值的像素值设为零 %
    x1=real(f1);x2=imag(f1);
    xx=find(x1<0);
    x1(xx(:))=0;
    f0__=x1;
    f0__([1:y/4],:)=0;f0__([(y/4)+y1-1:y],:)=0;
    f0__(:,[1:y/4])=0;f0__(:,[(y/4)+y1-1:y])=0;
    f0_=f0__;
    F0_=fft2(f0_);
```

```
    % 傅立叶域的条件限制 %
    H0=G.*(F0_.^(-1));
    h0=ifft2(H0);

    % 模糊的条件限制 %
    h0([1:y/4],:)=0;h0([(y/4)+y1-1:y],:)=0;
    h0(:,[1:y/4])=0;h0(:,[(y/4)+y1-1:y])=0;
    w1=real(h0);w2=imag(h0);
    ww=find(w1<0);
    w1(ww(:))=0;
    h0_=w1;
    H0_=fft2(h0_);
End

% 显示恢复后的图像 %
figure,imshow(real(f0_),[min(real(f0_(:))) max(real(f0_(:)))]);
figure,imshow(real(h0_),[min(real(h0_(:))) max(real(h0_(:)))]);
```

3. 结果展示



(a) 原始图像 $f(m,n)$　(b) PSF 图像 $h(m,n)$　(c) 受污染的图像 $g(m,n)$

(d) 猜测的初始图像　(e) 恢复的原始图像　(f) 恢复的 PSF 图像

图 L6.2-1

讨 论

实验结果显示，迭代盲目去卷积法虽然可以正确地恢复原始图像，但是此一方法有几点

需要注意：若初始图像为高灰度的图像，则恢复效果非常差；迭代中有许多参数是需要慎选的、否则并不确保迭代会收敛；此外噪声干扰的多寡也是图像迭代是否收敛的重要因素之一；另外，迭代的次数也是需要考虑的。因此，综合来说，我们必须常常从尝试错误中来寻得答案。

动手做

1. 调整不同的 PSF 图像的大小，重新执行此一方法，观察其迭代次数与恢复图像的关系。

第七章 实验——图像压缩

L7.1 游程长度编码

原 理

在实际的图像中，图像内的像素间都有相关性，因而产生可以去除的冗余性，达到图像压缩的目的。游程长度编码 (run length encoding, RLE) 即是直接利用此相关性的压缩方法之一，其基本做法是将一连串相同的数据以两个字节表示，一个代表字串的长度，另一个则表示数据。这一种编码方法对于一些特殊的图像，其压缩结果相当令人满意，例如小画家之类的绘图软件所产生的图形文件即属此类。

不过游程长度编码也有一个重大的缺点、那就是大多数相邻的像素重复的次数必须大于 2 才有压缩的效果，否则压缩后的数据可能比原始的数据还要多!这是使用游程长度编码最需要注意的一点。

程序范例

1. 目的

利用两组不同的数据，去验证游程长度编码的压缩效能。

2. 程序及说明

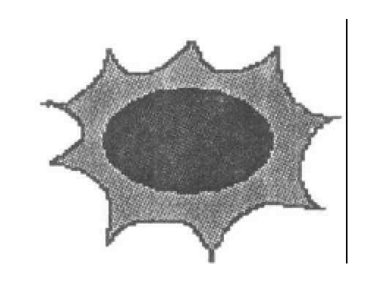
```
% 程序L7_1.m: 游程长度编码 %
% 取一个较不适合游程长度编码的图像作范例 %
[X,map]=bmpread('L7_1a.bmp');
X=X(100:220,100:220);
RLE(X,map)                        % 执行游程长度编码
%取另外一个适合做游程长度编码的图像再做一次编码%
[X,map]=bmpread('L7_1b.bmp');
figure
RLE(X,map)                        % 执行游程长度编码
% RLE.m
% 函数 RLE.m: 游程长度编码 %
function [CR,distortion]=RLE(X,map) % CR: 压缩率; distortion: 编解码的失真
Image1=X(1:100,1:100);            % X 为输入的图像
Image2=Image1';
```

```
image_1D=Image2(:);                    % 将原始图像写成一维的数据并设为
                                       % image_1D
nx=length(image_1D);
run_len(1)=1;
j=1;
for i= 1:1:nx-1;
  if image_1D(i)==image_1D(i+1);       % 将 image_1D 所包含的像素进行比对
  run_len(j)= run_len(j)+1;            % run_len (j)记录相对应的重复次数
  else
  data(j)=image_1D(i);                 % data (j)代表相对应的像素数据
  j=j+1;
  run_len(j)=1;
  end
end
data(j)=image_1D(nx);
r=1;
for j=1:length(run_len);               % 此循环的目的在于利用上述所得到的
  for s=1:1:run_len(j);                % 重复次数与对应到的数据，重建原始
  rec_1Dimag(r)=data(j);               % 数据，以验证此为无失真编码
  r=r+1;
  end
end
error=rec_1Dimag(:)-image_1D;
distortion=norm(error)
rec_1Dimag;
d=length(run_len);                     % 计算出run_len所占的字节数
CR=nx/(2*d);                           % 计算出全部数据经处理后是压缩
CR;                                    % 或者是膨胀
imshow(Image1,map)
```

3. 结果展示



(a) 不适合作游程长度编码的图像，CR = 0.646 4



(b) 适合作游程长度编码的图像，CR = 9.596 9

图　L7.1-1

distortion = 0 (表示无失真)

讨 论

由图 L7.1-1 的结果显示：两个特性不同的图像经过游程长度编码之后，有截然不同的结果。图 L7.1-1 (a) 的例子经过编码之后，数据量不但没有减少，反而增加了；而图 L7.1-1 (b) 的例子却有非常理想的效果。由此可知：游程长度编码较适用于某些特殊的图像，而非适用于所有的图像。且由 distortion = 0 也可验证游程长度编码为无失真编码。

动手做

1. 试试看用别的图像做游程长度编码，在做处理之前，先预测图像适不适合做游程长度编码，并验证其结果是否符合。

2. 试着以游程长度编码为基础，进一步改良其缺点，使新的编码方式能适合更多的图像。

L7.2 应用小波变换与向量量化做图像压缩

原 理

本实验应用小波变换分频理论，将一幅图像分成数个频带，再分别对其进行向量量化压缩。

由小波分频理论可知，越低频的系数包含的能量越大，对重建图像在视觉上较重要;相对的，越高频的系数所包含的能量越小，在视觉上则较不重要。所以我们在较低频带配置较多的压缩位，以得到较低的失真，而在较高的频带则配置较少的压缩位，以达到较高的压缩比，如此可兼顾压缩比与图像重建的品质。

本实验中向量量化码本均依照 L3.2 节的 LBG 算法准则来加以设计。

程序范例

1. 目的

将一幅图像经过一层小波变换分频成 LL1、LH1、HL1、HH1 四个频带，再分别利用 LBG 训练其不同频带的码本 cbLL1、cbLH1、cbHL1、cbHH1，最后分别进行向量量化。

2. 程序及说明

```
% 程序 L7_2.m: 应用小波变换与向量量化做图像压缩 %
% 利用 LBG 训练一层小波变换后四个频带的码本 %
load lena
[LL1,LH1,HL1,HH1]=dwtper2(X,'bior3.3');
```

```
% 训练 LL1 频带码本，将码本大小设为 128，维度为 4 %
th=0.05;
nc=128; nd=4;
[cbLL1]=LBG(LL1,nc,nd,th); % LBG.m 在 L3.3 节中
% 训练 LH1 频带码本，将码本大小设为 64，维度为 4 %
nc=64; nd=4;
[cbLH1]=LBG(LH1,nc,nd,th);
% 训练 HL1 频带码本，将码本大小设为 64，维度为 4 %
nc=64; nd=4;
[cbHL1]=LBG(HL1,nc,nd,th);
% 训练 HH1 频带码本，将码本大小设为 64，维度为 16 %
nc=64; nd=16;
[cbHH1]=LBG(HH1,nc,nd,th);

% 利用一层小波变换加上向量量化进行 Lena 图像压缩 %
% 输入一幅 Lena 图像 %
load lena
[M,N]=size(X);
% 使用 Matlab 中 Wavelet 工具箱的周期性小波变换函数 dwtper2.m %
% 并使用双正交小波 bior3.3 进行一层小波变换 %
[LL1,LH1,HL1,HH1]=dwtper2(X,'bior3.3');
wt_X(1:M/2,1:N/2)=LL1;
wt_X(1:M/2,N/2+1:N)=LH1;
wt_X(M/2+1:M,1:N/2)=HL1;
wt_X(M/2+1:M,N/2+1:N)=HH1;
% 分别针对不同的频带进行向量量化，其中 VQ.m 置于 L3.3 节中 %
[R_LL1,bitLL1]=VQ(LL1,cbLL1);
[R_LH1,bitLH1]=VQ(LH1,cbLH1);
[R_HL1,bitHL1]=VQ(HL1,cbHL1);
[R_HH1,bitHH1]=VQ(HH1,cbHH1);
wt_Y(1:M/2,1:N/2)=R_LL1;
wt_Y(1:M/2,N/2+1:N)=R_LH1;
wt_Y(M/2+1:M,1:N/2)=R_HL1;
wt_Y(M/2+1:M,N/2+1:N)=R_HH1;
% 计算图像经过向量量化后的压缩比 CR 及失真 PSNR %
totalbit = bitLL1+bitLH1+bitHL1+bitHH1;        % 花费的总位数
Y=idwtper2(R_LL1,R_LH1,R_HL1,R_HH1,'bior3.3'); % 反小波变换
```

```
MSE=(sum(sum((X-Y).^2)))/(M*N);
PSNR=20*log10(255/sqrt(MSE))
CR=M*N*8/totalbit
```

3. 结果展示



(a)原始 Lena 图像



(b) 经一层小波变换后所得的分频图像，由左上、右上、左下、右下分别为 LL1、LH1、HL1 以及 HH1



(c) 经向量量化后所得的分频图像还原图，由左上、右上、左下、右下分别为 R_LL1、R_LH1、R_HL1 以及 R_HH1



(d) 经小波反变换后所得的重建图像

图 L7.2-1 利用一层小波变换加向量量化进行图像压缩的结果

讨 论

由 L3.3 及 L3.4 节可知，码本的大小越小或维度越大，其训练集与码本的平均误差越大。另外由图 L7.2-1 (b) 可以看出，经过小波变换后，越低频的系数变动范围越大，故在设计码本时，对较低频带采用大小较大且维度较小的码本进行向量量化。

又从能量的观点来说，越低频系数所包含的能量越多，对图像还原的影响极大，因此给予较高的传输位。

本实验的结果，各频带向量量化的压缩比分别为 CR_LL1=4.6、CR_LH1=5.33、CR_HL1=5.33 及 CR_HH1=21.33，最后图像重建的结果为 PSNR=32，CR=9.14。

动手做

1. 请改以小波基底“db8”来进行小波变换，亦即将程序中的“bior3.3”改成“db8”，观察其结果的差异。

2. 试着将 cbLL1 的大小设成 128，维度变成 1；cbLH1 及 cbHL1 不变；cbHH1 的大小设为 32，将维度设为 16，并观察其结果与先前的差异。

第八章　实验——图像分割

L8.1　像素聚类区域成长法

原　理

顾名思义，此方法从一种子像素开始，通过如平均灰度、组织纹理及色彩等性质的判断，将具类似性质的像素逐一纳入所考虑区域中，使此区域由种子逐渐成长成一个性质相似的图像区块。

最简单的做法是以一个种子像素为基础，并设定一个像素差临界值，若邻近像素差值小于或等于此临界值，则将此邻近像素纳入此区块内。借助此方法的不断向外迭代聚类区域成长，可得到像素值相似的图像区块。

程序范例

1. 目的

将像素差临界值设为 0，随意取一种子点，观察其像素聚类区域成长的结果。

2. 程序及说明

```
% 程序 L8_1.m: 像素聚类区域成长法 %
% 读入一幅区域明显的图像，实验像素聚合 %
[X,map]=bmpread('L8_1.bmp');
seed=[250 200];                       % 随意设定一个种子点
[Y_1]= regiongrow(X,seed,0,1000);     % regiongrow 程序在本实验范例中
[Y_2]= regiongrow(X,seed,0,10000);
[Y_3]= regiongrow(X,seed,0,inf);
figure
subplot(221),imshow(X,map)
subplot(222),imshow(Y_1,map)
subplot(223),imshow(Y_2,map)
subplot(224),imshow(Y_3,map)

% 函数 regiongrow.m: 像素聚类区域成长法 %
```

```
function [O_image]= regiongrow(I_image,seed,region_th,stopn)
% [O_image]= regiongrow(I_image,seed,region_th,stopn) %
% I_image : (row x col) 输入图像 %
% seed : [yseed xseed] 输入一聚合种子的坐标 %
% region_th : (0<= region_th <=255) 区域临界值 %
% stopn : (0<stopn<=inf)中断临界值:控制中断程序的聚合像素个数 %
% O_image : (row x col) 经过像素聚类区域成长后的输出图像 %

O_image=I_image;
[row,col]=size(O_image);
% 将种子放人队列中 %
yseed=seed(1);
xseed=seed(2);
pixel_seed=I_image(yseed,xseed);   % 种子的像素值
queue=[yseed xseed];                % 将种子的坐标存放人队列中
top=1;                              % 设定队列矩阵的顶端位置

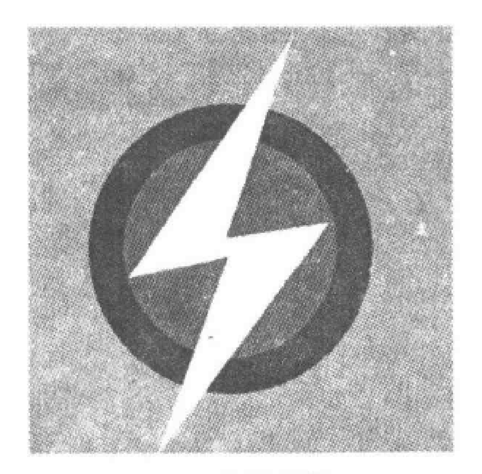
% 制造一个标注像素聚合的矩阵 %
region_matrix=zeros(row,col);
region_matrix(yseed,xseed)=1;       % 若判定为相同区域，则标注为 1，反之为 0

% 开始像素聚合 %
region_count=1;           % 制造一个聚合区域的计数器
while top~=0              % 当队列矩阵中没有存放任何坐标时，则停止循环
  % 取出队列矩阵中的最底端坐标值 %
  row_buttom=queue(1,1);
  col_buttom=queue(1,2);
  pixel_buttom=I_image(row_buttom,col_buttom);
  % 以此坐标轴为种子，判断周围各像素是否属于相同的区域 %
  regionedge=0;
 % 设一个判断是否为区域边缘的指标 %
  for i=-1:1
    for j=-1:1
        if row_buttom+i<=row & row_buttom+i>0 &...
              col_buttom+j<=col & col_buttom+j>0
```

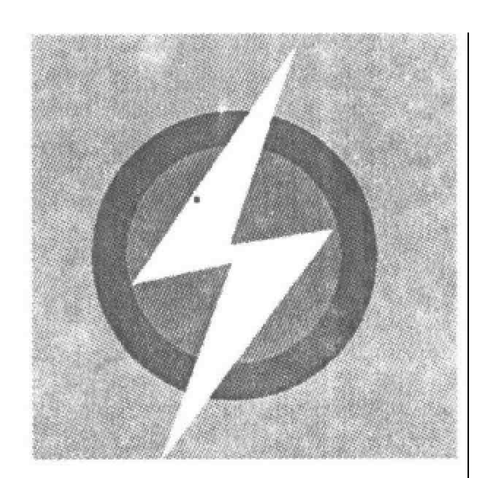
```
            if abs(I_image(row_buttom+i,col_buttom+j)-pixel_buttom)...
                <=region_th & region_matrix(row_buttom+i,col_buttom+j)~=1
                 % 若像素之间的差值小于区域临界值，则判定为同一区域 %
                 top=top+1;            % 队列矩阵顶端 top 值增加 1
                 % 将此像素坐标存入队列矩阵的顶端 %
                 queue(top,:)=[row_buttom+i col_buttom+j];

                 region_matrix(row_buttom+i,col_buttom+j)=1;% 标注为 1
                 region_count=region_count+1;
                 % 将放入队列的像素值均先内设为 0 %
                 O_image(row_buttom+i,col_buttom+j)=0;
              end
              if region_matrix(row_buttom+i,col_buttom+j)==0
                 regionedge=1;             % 判断为区域边缘
              end
            else
              regionedge=1;                % 判断为区域边缘
            end
        end
    end
    % 将判断为非聚合区域边缘的像素值用种子的像素值取代 %
    if regionedge~=1
        O_image(row_buttom,col_buttom)=I_image(yseed,xseed);
    end
    % 若计数值大于或等于所设定的中断临界值，则将不再进行聚合循环 %
    if region_count>=stopn
        top=1;
    end
    % 将已经做完聚合的坐标从队列矩阵中删去 %
    queue=queue(2:top,:);
    top=top-1;
end
O_image(yseed,xseed)=0;            % 将种子像素值设为 0，即黑色
```

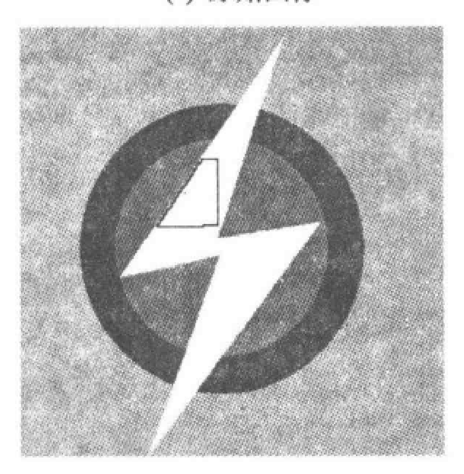
3. 结果展示



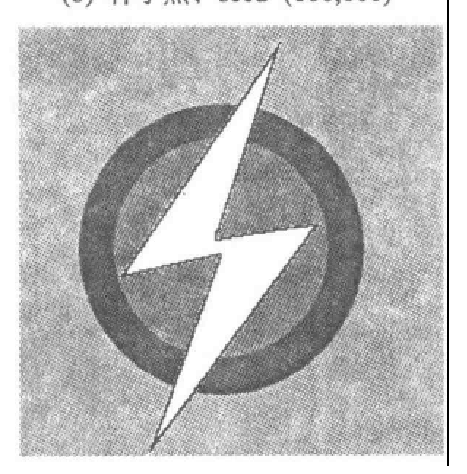
(a) 原始图像



(b) 种子点，seed=(100,100)



(c) 设定当聚类区域成长的像素个数达 1 000 时，所得的图像、其中黑线为此区域的边缘线



(d) 成长完成的结果

图 L8.1-1　像素聚类区域成长过程图

讨　论

从图 L8.1-1 可以看出，像素聚合方法可以成功地找到像素性质相似的区域，并将其边界找出，因此，此方法极适用于检测边缘明显的图像区块。

必须要注意的是判断邻近像素是否属于同一区块的临界值的选定极为重要，若选得过大，则像素过度聚合，反之，若临界值选择过小，则聚合不够，两者都无法忠实反映出该图像区块的真实形状。

动手做

1. 请另外随意输入一个种子点坐标，观察其像素聚合的过程。

2. 利用一幅不同于本实验所采用的图像，重复上题的实验，并尝试加大判断邻近像素是否属于同一区块的临界值，观察其结果的差异。

L8.2 四叉树区域分割与合并法

原 理

首先将图像分割成不重叠的区域，其中最常用的方法是分割成四个大小相同的子图像。若每个子图像内有性质不同的图像存在就将该子图像继续分割，持续这个过程直到没有可再分割的条件为止。接着将具同性质又相邻的子区域进行合并，直到无法再合并为止，整个图像分割程序才算大功告成。

程序范例

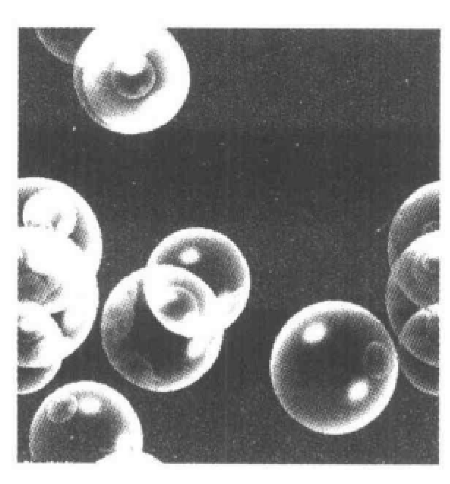
1. 目的

使用四叉树 (quadtree) 的方法，来找出图像中物体的轮廓。

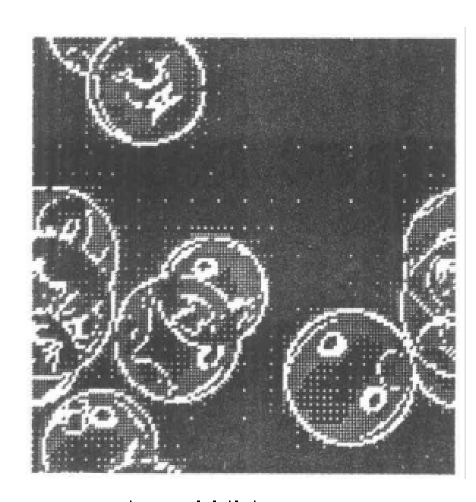
2. 程序及说明

```
% 程序 L8_2.m: 四叉树区域分割与合并法 %
image1=imread('L8_2.bmp'); % 读取图像
s=qtdecomp(image1,0.1);    % qtdecomp 就是“四叉树分解”函数
                           % 0.1 是每个方块所需达到的最小差值
image2=full(s);            % 由于 s 变量为 sparse (稀疏) 矩阵，故我们使用 full
                           % 指令来将 s 矩阵中缺少的 element 补上，以便可
                           % 用图像显示出来，新的矩阵值存于 image2 中
imshow(image1);            % 将原图像显示出来
figure,imshow(image2)      % 将四叉树分解处理过后的图像显示出来
```

3. 结果展示



(a) 原图像



(b) 经过四叉树分解处理过后的结果

图 L8.2-1

讨　论

图 L8.2-1 的独立亮点都是四叉树分割块中左上角的位置，由此可体会出此方法的中间结果。最后可将独立亮点视为噪声，用简单的后处理方法 (例如：中值滤波) 将其删除。

动手做

1. 改变每区块所需达到的最小差值，对处理结果有何影响?
2. 对一个 8 位的灰度图像 image、s = qtdecomp(image,0.1) 中的 0.1 代表实际上灰度差是多少?

L8.3 Sobel 边缘检测

原　理

对于一张暗色背景下有亮条纹的图像、由水平扫描线的灰度剖面图及其一阶导数图可追现一个边缘，这就是边缘检测的基本理论。图像 $f(x,y)$ 的一阶偏导数与梯度有关，即 $\nabla f=\frac{\partial f}{\partial x}\boldsymbol{i}+\frac{\partial f}{\partial y}\boldsymbol{j}=G_x\boldsymbol{i}+G_y\boldsymbol{j}$。梯度向量的方向是 f 在(x,y) 处变化最大的方向，在边缘检测中一个最重要的量是这个向量的大小，即 $|\nabla f|=\sqrt{G_x^2+G_y^2}$ 、为计算方便通常近似为

$$|\nabla f|=|G_x|+|G_y| \tag{L8.3-1}$$

对 Sobel 运算而言

$$G_x=(Z_7+2Z_8+Z_9)-(Z_1+2Z_2+Z_3) \tag{L8.3-2a}$$

$$G_y=(Z_3+2Z_6+Z_9)-(Z_1+2Z_4+Z_7) \tag{L8.3-2b}$$

其中 $Z_1,Z_2,\cdots,Z_9$ 的位置请参考图 L8.3-1。

Z_1	Z_2	Z_3
Z_4	Z_5	Z_6
Z_7	Z_8	Z_9

图 L8.3-1　3×3 图像方块

对算出的梯度大小取一临界值，大于临界值的像素设为 1，表示边缘像素、反之为 0，表示背景像素。

程序范例

1. 目的

给一张图像以 Sobel 算法进行边缘检测。

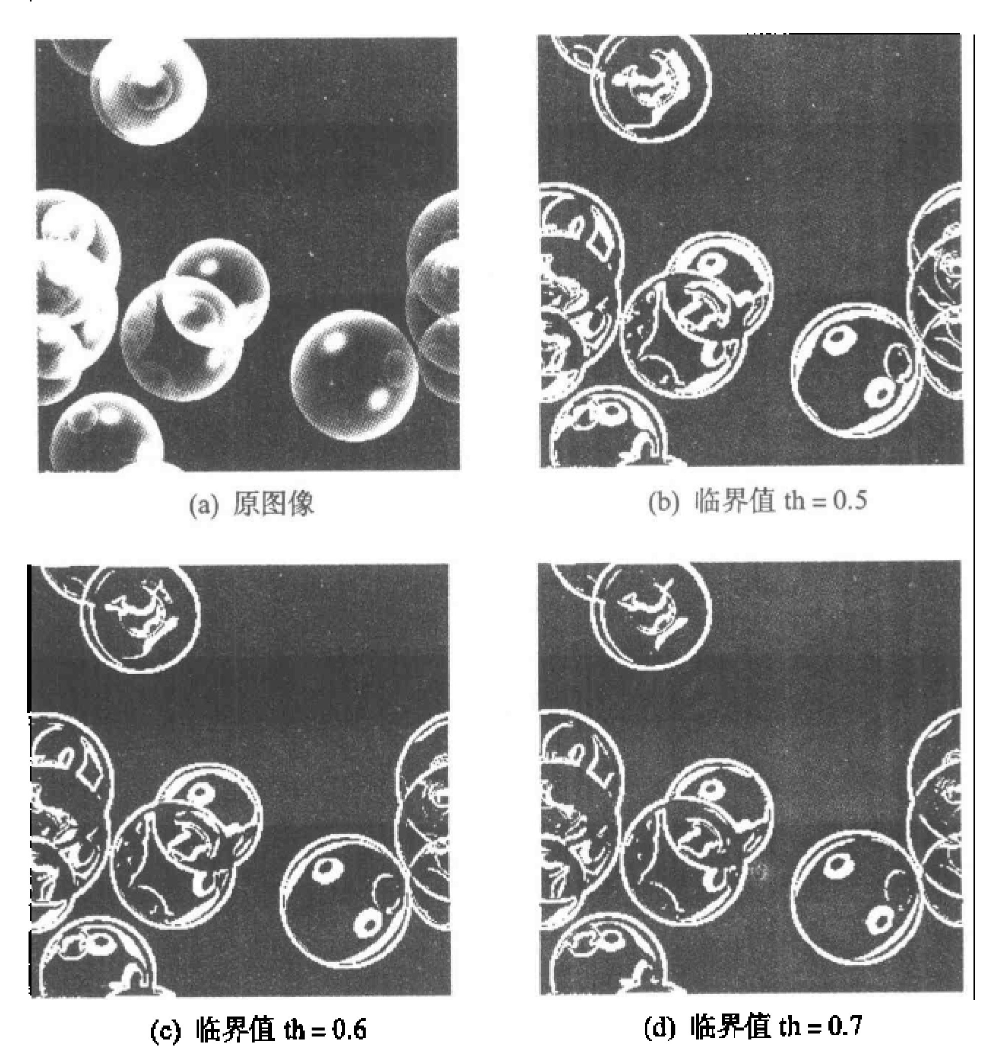
2. 程序及说明

```
% 程序 L8_3.m: Sobel 边缘检测 %
[image,map]=bmpread('L8_2.bmp');
image1=sobel(image,0.5);
image2=sobel(image,0.6);
image3=sobel(image,0.7);
imshow(image,map)
figure,imshow(image1)
figure,imshow(image2)
figure,imshow(image3)

% 函数sobel.m: sobel算子 %
function  y=sobel(image,th)
% image: 输入图像%
% th: 临界值%
[Nn  Nm]=size(image);                % 图片大小
h=[-1  -2  -1;0  0  0;1  2  1];      % Sobel 算子
Gx=filter2(h,image);                 % 得梯度向量的Gx分量
Gy=filter2(h',image);                % 得梯度向量的Gy分量
F=abs(Gx)+abs(Gy);
Bw_Gx=double(Gx)/255;                % 将Gx范围从[0, 255]缩到[0,1]
Bw_Gy=double(Gy)/255;                % 将Gy范围从[0, 255]缩到[0,1]
Bw_F=double(F)/255;                  % 将F范围从[0, 255]缩到[0,1]
                                     % 设定临界值，在[0,1]之间
for  i=1:Nn
   for  j=1:Nm
      if  Bw_F(i,j)<th
         y(i,j)=0;
      else
         y(i,j)=1;
```

```
      end
    end
  end
```

3. 结果展示



(a) 原图像　(b) 临界值 th = 0.5

(c) 临界值 th = 0.6　(d) 临界值 th = 0.7

图　L8.3-1

讨　论

若要设定临界值，首先要看直方图的分布，看看像素值的分布，寻找其分界的地方为临界值，以本图像为例，差不多取到 0.5，其边缘效果就已经很明显了。

动手做

1. 对上例的原图像作直方图均衡化，并重取一个临界值，作 Sobel 边缘检测，使其边缘效果更明显。目的可达到吗？为什么？

2. 分析临界值和边缘效果的关联性。

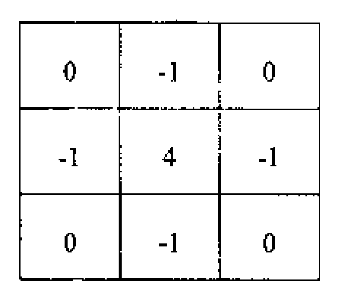
L8.4　拉氏边界检测法

原　理

二维图像函数 $f(x,y)$ 的拉式算子是一个二阶导数，其基本要求是与中心像素点相对应的模板系数必须为正，与外围的像素点相对应的系数必须为负。对于一个 3×3 的区域，在实际应用中最常用的计算方式是

$$\nabla^2 f = 4Z_5 - (Z_2 + Z_4 + Z_6 + Z_8) \tag{L8.4-1}$$

图 L8.4-1 是实现 (L8.4-1) 式的空间模板。



0	-1	0
-1	4	-1
0	-1	0

图 L8.4-1　实现拉式运算的空间模板

尽管拉式算子对于亮度转变(例如边缘)很敏感，但却很少用于实际的边缘检测，因为它对噪声有不可接受的敏感性，会产生双边缘并且不能检测出边缘的方向。要降低噪声的影响，可先做平滑处理。而最好的选择是具有 Gaussian 脉冲响应的低通滤波器，其函数形式是

$$h(x,y) = \exp\left(\frac{-x^2 + y^2}{2\sigma^2}\right) \tag{L8.4-2}$$

其中 σ 值是标准差，令 $r^2 = x^2 + y^2$，h 对 r 取二阶导数得 (也是 h 的 Laplacian 算子) 得

$$\nabla^2 h = \left(\frac{r^2 - \sigma^2}{\sigma^4}\right)\exp\left(\frac{-r^2}{2\sigma^2}\right) \tag{L8.4-3}$$

就成了拉氏—高斯滤波器的响应。

程序范例

1. 目的

对一张图像以二维的拉氏—高斯算子来显示图像的边缘。采用不同的 σ 值来运算，并比较其差异。

2. 程序及说明

```
% 程序 L8_4.m: 拉氏边界检测法 %
[image map]=bmpread('L8_4.bmp');   % 读取图像
image_edge_1=edge(image,'log',[],2); % 使用边缘检测中的越零点检测，其中[ ] 是空矩阵，
                                     % 其 edge 函数会自动检测最敏感的临界值，2 是σ值
image_edge_2= edge(image,'log',[],4); % 同上，不过σ值为 4
imshow(image,map)                    % 显示原图像
figure,imshow(image_edge_1)          % 同时显示σ值为 2 的图像
figure,imshow(image_edge_2)          % 同时显示σ值为 4 的图像
```

3. 结果展示

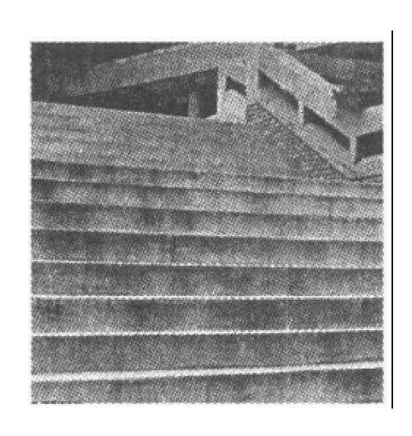
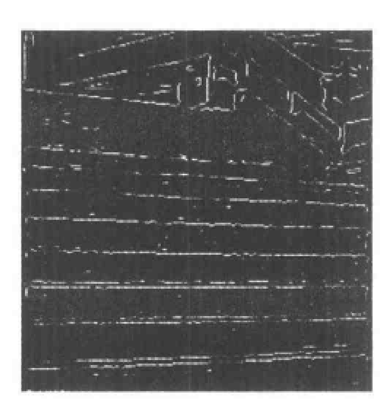
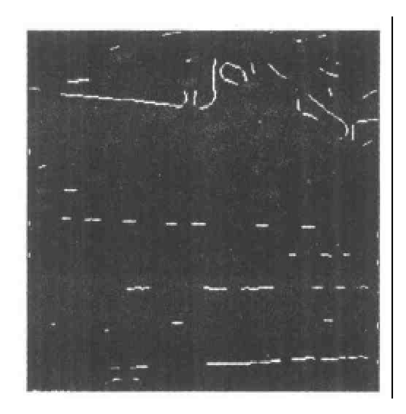


图 L8.4-1 原图像



(a) $\sigma=2$ 的结果



(b) $\sigma=2$ 的结果

图 L8.4-2 以拉氏—高斯算子及越零检测作边缘检测

讨 论

要使图像的亮度变化剧烈且噪声相对要低，梯度算的边缘检测才可以做得很好。此外，由越零点可得一个可靠的边缘位置，且 $\nabla^2 h$ 的平滑特性降低噪声的影响，同时使图像模糊的程度正比于 σ。

动手做

1. 证明由 (L8.4-3) 式所给的拉式算子 $\nabla^2 h$ 的平均值为零。
2. 在原图像上加入高斯或胡椒盐的噪声，以范例的程序处理，并讨论其结果。

L8.5 LUM 滤波器

原 理

关于 LUM 滤波器的原理请参考本书的 8.4 节。值得注意的是平滑参数 k 越大，平滑的效果越明显；又当锐化参数 l 值越小，锐化程度越高。借助适当的参数选取，LUM 滤波器可同时当做平滑滤波器及锐化滤波器。

程序范例

1. 目的

将一幅图像加入噪声后，分别以 $(k,l)=(3,6)$以及$(9,12)$做 LUM 滤波，并观察其结果。

2. 程序及说明

```
% 程序L8_5.m: LUM滤波器 %
% 对一幅加噪声的图像，实验LUM滤波器 %
load lena
Subplot(221),imshow(X,map)

% 将原始图像加入噪声 %
X=imnoise(X,'salt & pepper',0.05);          % 加入密度为0.05的噪声
Subplot(222),imshow(X,map)

% 进行LUM滤波 %
[Y_k3l6]=LUM(X,3,6,5);                      % LUM函数在本范例中
Subplot(223),imshow(Y_k3l6,map)

[Y_k9l12]=LUM(X,9,12,5);
Subplot(224),imshow(Y_k9l12,map)
```

```
% 函数 LUM.m: LUM 滤波器 %
function [O_data]=LUM(I_data,k,l,w)
% w=input('please input window size (5 or 7):') 模板大小 %
% k=input('please input k parameter:') 平滑参数 %
% l=input('please input l parameter:') 锐化参数 %

I_data1=I_data;
w1=(w-1)/2;
% 将原始图像依照模板的尺寸扩大 %
[M,N]=size(I_data);
A=[];
A=[I_data(:,1:w1) I_data I_data(:,N-w1+1:N)];
A=[A(1:w1,:);A;A(M-w1+1:M,:)];
I_data=A;

% 进行 LUM 滤波 %
for i=1:1:M
    for j=1:1:N
        block=I_data(i:i+w-1,j:j+w-1);     % 取一个 W×W 图像模板进行滤波
        b=sort(block(:));                  % 将此模板的内容依照大小排序
        tl=(b(l)+b(w^2-l+1))/2;            % 找出中间值
        % 决定此模板中心新的数值 %
        if I_data(i+w1,j+w1)<b(k)
            O_data(i,j)=b(k);
        elseif I_data(i+w1,j+w1)>b(l) & I_data(i+w1,j+w1)< = tl
            O_data(i,j)=b(l);
        elseif I_data(i+w1,j+w1)>tl & I_data(i+w1,j+w1)<b(w^2-l+1)
            O_data(i,j)=b(w^2-l+1);
        elseif b(w^2-k+1)<I_data(i+w1,j+w1)
            O_data(i,j)=b(w^2-k+1);
        else
            O_data(i,j)=I_data(i+w1,j+w1);
        end
    end
end
```

3. 结果展示



(a) 原始 Lena 图像　(b) 加噪声后的图像

(c) 经 (k,l) = (3,6) 的 LUM 滤波后的图像　(d) 经 (k,l) = (9,12) 的 LUM 滤波后的图像

图 L8.5-1　经 LUM 滤波后的结果

讨　论

从图 L8.5-1 可以得知，当平滑参数 k 设得愈大时，LUM 滤波器滤除噪声的效果也就愈好，但图像中一些变化较剧烈的部分，却变的较为模糊。反之当锐化参数 l 设的较小时，图像的对比度较明显，却也可能放大噪声的效应。因此我们可以用互动式的方法，适当巧妙地运用这两个参数，以获得我们需要的结果。

动手做

1. 尝试改变 LUM 滤波器平滑参数及锐化参数，观察其经过滤波后的图像。

2. 改变原始图像所加入噪声的密度，并利用相同的参数设定进行 LUM 滤波，讨论与本实验结果的差异。

第九章　实验——表示与描述

L9.1　不变矩

原　理

一个与平面区域有关的几何特性，像是大小、位置、方向及形状等，其中很多特性与矩这个参数有关。一阶矩与形状有关；二阶矩显示曲线围绕直线平均值的扩展程度；三阶矩则是关于平均值的对称性的测量。由二阶矩与三阶矩可以导出一组共七个不变矩，这一组不变矩不受平移、旋转和比例变化的影响。

程序范例

1. 目的

给一张简单的几何图形图像，分别做尺寸大小和旋转角度的变化，观察其七个不变矩的变化。

2. 程序及说明

```
% 程序L9_1.m: 不变矩 %
[image map]=bmpread('L9_1.bmp');
invariancematrix1= invariance(image,1,0);      % 原始图像的不变矩
invariancematrix2= invariance(image,0.5,0);    % 一半尺寸的不变矩
invariancematrix3= invariance(image,1,90);     % 旋转90°的不变矩
invariancematrix4= invariance(image,1,180);    % 旋转180°的不变矩
invariancematrix1=abs(invariancematrix1)
invariancematrix2=abs(invariancematrix2)
invariancematrix3=abs(invariancematrix3)
invariancematrix4=abs(invariancematrix4)

% 函数invariance.m: 求不变矩 %
function [invariancematrix]=invariance(image,size_order,rotate)
% image: 输入图形 %
% size_order: 图形的放大倍率 %
% rotate: 图形旋转的角度 %
```

```
% invariancematrix: 计算出的不变矩 %

X=image;
[F_x F_y]=size(X);
F=IMRESIZE(X,[fix(F_x*size_order) fix(F_y*size_order)]);     % 改变图像大小
F=imrotate(F,rotate);                                         % 改变图像角度

[F_x F_y]=size(F);

% 求x的平均值 %
F_xsum=sum(F,2);
temp=1:F_x;
temp=temp';
mean-_temp=temp.*F_xsum;
mean_x=sum(mean_temp)/sum(sum(F));

% 求y的平均值 %
F_ysum=sum(F,1);
temp=1:F_y;
mean_temp=temp.*F_ysum;
mean_y =sum(mean_temp)/sum(sum(F));

% 求各阶矩 %
for p=0:3
    for q=0:3
        r(p+1,q+1)=(p+q)/2+1;
        x=1:F_x;
        x=x-mean_x;
        x=x.^p;
        y=1:F_y;
        y=y-mean_y;
        y=y.^q;
        u_temp=x'*y;
        u(p+1,q+1)=sum(sum(u_temp.*F));
    end
end
```

```
% 标准中心矩 %
n=u./u(1,1).^r;

% 七个不变矩 %
invariancematrx(1,1)=log10(abs(n(3,1)+n(1,3)));
invariancematrx(1,2)=log10(abs((n(3,1)-n(1,3)).^2+4*n(2,2).^2));
invariancematrx(1,3)=log10(abs((n(4,1)-3*n(2,3)).^2+(3*n(3,2)-n(1,4)).^2));
invariancematrx(1,4)=log10(abs((n(4,1)+n(2,3)).^2+(n(3,2)+n(1,4)).^2));
invariancematrx(1,5)=log10(abs((n(4,1)-3*n(2,3))*(n(4,1)+n(2,3))*((n(4,1)-n(2,3)).^2-3*(n(3,2)+
n(1,4)).^2)+(3*n(3,2)-n(1,4))*(n(3,2)+n(1,4))*((3*n(4,1)+n(2,3)).^2-(n(3,2)+n(1,4)).^2)));
invariancematrx(1,6)=log10(abs((n(3,1)-n(1,3))*((n(4,1)+n(2,3)).^2-(n(3,2)+n(1,4)).^2)+4*n(2,2)*
(n(4,1)+n(2,3))*(n(3,2)+n(1,4))));
invariancematrx(1,7)=log10(abs((3*n(3,2)-n(1,4))*(n(4,1)+n(2,3))*((n(4,1)+n(2,3)).^2-3*(n(3,2)+
n(1,4)).^2)+(3*n(2,3)-n(4,1))*(n(3,2)+n(1,4))*((3*n(4,1)+n(2,3)).^2-(n(3,2)+n(1,4)).^2)));
```

3. 结果展示



(a) 原始 (b) 一半尺寸 (c) 旋转 90° (d) 旋转 180°

图 L9.1-1 简单的几何图形

讨 论

因为所得的不变矩动态范围很大，所以取 $\log|\phi_i|$ 。另外，为了方便起见取正值。由表 L9.1-1 所列结果显示，所得的七个不变矩很相似，验证了不变矩的确是不受旋转及大小比例改变的影响。表 L9.1-1 中同一行函数值间的误差主要来自于数据的离散化特性以及计算精度的效应。

表 L9.1-1 三种变化的不变矩

不变矩 $\|\log\|\phi_i\|\|$	$i=1$	$i=2$	$i=3$	$i=4$	$i=5$	$i=6$	$i=7$
原 始	1.046 9	6.037 1	7.743 7	6.032 4	13.097 4	9.055 2	13.074 6
一半尺寸	1.047 1	6.002 4	7.688 8	6.022 8	13.057 4	9.028 1	13.043 0
旋转 90°	1.046 9	6.037 1	7.743 7	6.032 4	12.774 4	9.055 2	12.991 9
旋转 180°	1.047 1	6.033 6	7.579 6	6.036 0	12.976 3	9.055 8	13.059 1

因为不变矩不受旋转及大小的影响，我们可以将其利用于识别二维或三维物体。不过这些不变矩并不足以区别所有的形状，而且对噪声很敏感。

动手做

1. 试随意用一幅图像，利用上述程序求出经过不同角度及尺寸的变化后的七组不变矩，看看七个不变矩的值是否相同?

2. 试写出一幅图像经任意平移后的七组不变矩，看看七个不变矩的值是否相同?

L9.2　细　化

原　理

在光学文字识别中，细化可以保留文字的信息，并消除多余的数据量。经细化后的文字，用各种表示法和描述法来对这些骨架字体做编码的工作比较方便。在印刷字、手写字的识别及光学干涉条纹的辨认算法中，每一笔划或条纹的宽度，对于辨认的工作并无帮助，反而造成辨认上的困难。因此使用细化将不等宽度的字体或条纹细化成等宽度的像素连线后，有助于进一步辨认的工作。有关骨架的定义及本实验所提的细化算法，请参考本书 9.1-4 节。另外，也可利用 Matlab 中 erode 函数中 “thin” 的功能来进行细化的过程。

程序范例

1. 目的

给一张简单的几何图形图像，分别做尺寸大小和旋转角度的变化，观察其七个不变矩的变化。

2. 程序及说明

```
% 程序 L9_2.m: 细化 %
[image,map]=bmpread('L9_2.bmp');          % image 像素值为 1 或 2
image=image-1;                             % 原图像素值设为 1 或 0
x=double(image);                           % 将图像变换为 double 格式
check=0;
while check==0
   [y count1]=thin(x,1);                   % type =1:做一次东南边界的细化
   [x count2 ]=thin(y,2);                  % type =2:做一次西北边界的细化
   if (count1&count2)==0                   % count 1=0, count2=0:表示细化完成
      check=1;
   end
end
y=x;
```

```
y1=logical(uint8(round(y)));          % 将图像变换为 unit8 的格式
imshow(y1)                            % 显示图形

% 函数 thin.m: 细化 %
% type=1:做一次东南边界的细化 %
% type=2:做一次西北边界的细化 %
% x:输入图像 %
% y:输出图像 %
% count =1:已做细化 %
% count =0:没有做细化 %

function [y,count]=thin(x,type)
  [xi,xj]=size(x);                    % 图像大小
  mask=zeros(xi,xj);
  for i=2:(xi-1)
    for j=2:(xj-1)
      if x(i,j)~=0
        s=0;p1=x(i,j);p2=x(i,j-1);p3=x(i+1,j-1);p4=x(i+1,j);
        p5=x(i+1,j+1);p6=x(i,j+1);
        p7=x(i-1,j+1);p8=x(i-1,j);p9=x(i-1,j-1);
        b=[p2 p3 p4 p5 p6 p7 p8 p9 p2];[nn nm]=size(b); n=sum(b(1:8));

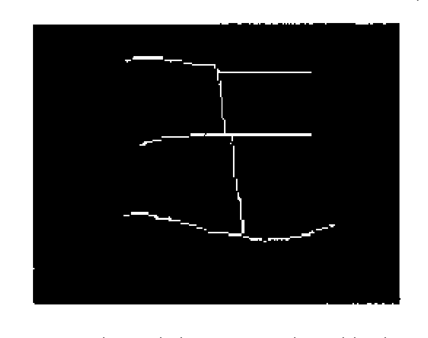
        for u=1:nm-1
          if (b(u+1)-b(u))<0
            s=s+1;
          end
        end
        if type==1
           l1=p2*p4*p6;  % 东边界的判别
           l2=p4*p6*p8;  % 南边界的判别
        elseif type==2
           l1=p2*p4*p8;  % 北边界的判别
           l2=p2*p6*p8;  % 西边界的判别
        end
        if (2<=n & n<=6 & s==1 & l1==0 & l2==0)
           mask(i,j)=p1;  % 满足上式则 p1=0
           count=1;
```

```
        else
          count=0;
        end
      end
    end
  end
y = x-mask;
```

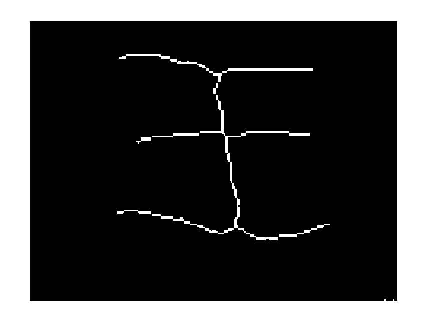
3. 结果展示



(a) 原图像



(b) 直接执行细化程序的结果



(c) 利用 erode 函数中'thin' 的功能将原图像腐蚀 15 次的结果

图 L9.2-1　图像细化

讨　论

由图 L9.2-1 可发现当用 erode 函数库里的 thin 来腐蚀以达到细化的效果时，在文字交叠的部分会产生线条扭曲。这是因为在前几次腐蚀的时候，在文字交叠的部分较为肥大之故。依照细化方法写出的程序较为方便，不用知道需要使用几次，便可得到结果，而且没有线条扭曲的现象。

动手做

1. 试只以东南方向进行细化，比较其图形与图 L9.2-1 (b) 的差异。
2. 试只以西北方向进行细化，比较其图形与图 L9.2-1 (b) 的差异。

L9.3　膨胀和腐蚀

原　理

腐蚀：对于整数构成的集合 B 和 S 而言，B 被 S 腐蚀，用 $B\ominus S$ 表示，定义为 $B\ominus S=\{x\,|\,(S)_x\subseteq B\}$，也就是被 S 腐蚀的 B 是 S 平移 x 之后包含在 B 中的所有点 x 的集合。

膨胀：若 B 被 S 膨胀，记为 $B\oplus S$，定义为 $B\oplus S=\{x\,|\,(\hat{S})_x\cap B\neq\varnothing\}$，其中 $\hat{S}$ 为 S 的映像或反射，也就是说将 S 对它的原点进行反射；B 被 S 的膨胀是所有可以使 S 的反射在平移 x 后与 B 仍有非空交集的 x 的集合。

程序范例

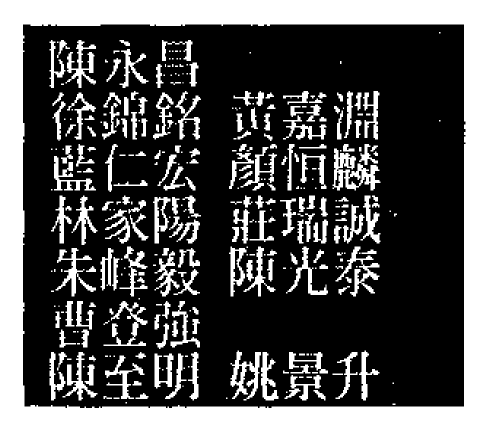
1. 目的

了解膨胀与腐蚀对图像的实际效应。

2. 程序及说明

```
% 程序 L9_3.m: 膨胀与腐蚀 %
image1=imread('L9_3.bmp');      % 读取原图像
S=ones(3,1);                    % 设 S 为一个 3×1 的结构元素
image2=erode(image1,S);         % 将原图像用 S 做腐蚀
image3=dilate(image1,S);        % 将原图像用 S 做膨胀
imshow(image1)                  % 显示原图像
figure,imshow(image2)           % 显示已腐蚀的图像
figure,imshow(image3)           % 显示已膨胀的图像
```

3. 结果展示




(a) 原图像

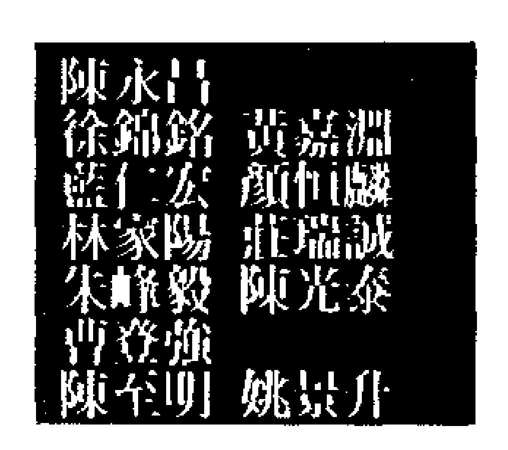
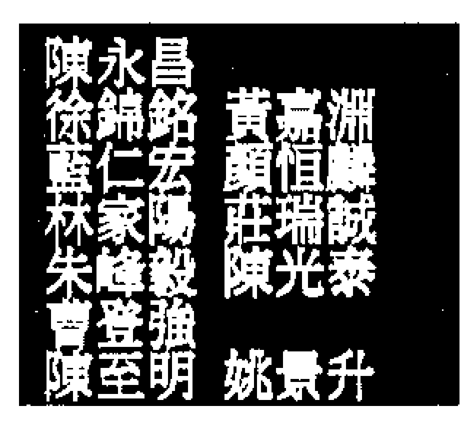

(b) 腐蚀的结果

图　L9.3-1 (待续)



(c) 膨胀的结果

图　L9.3-1 (续)

讨　论

由图 L9.3-1 可看出，膨胀使字形变宽，腐蚀使其变窄，完全符合我们的直觉。

动手做

1. 设计 5×1 的结构元素 S，再对图像做膨胀与腐蚀，并比较其结果。
2. 设计 1×1 的结构元素 S，再对图像做膨胀与腐蚀，并比较其结果。

L9.4　断开与闭合

原　理

断开 (Opening) 通常是平滑图像轮廓，截断窄的细颈，消除细的分支。闭合 (Closing) 也倾向于平滑轮廓部分，不过与断开是相反的。一般来说，它把窄的中断部分和长的细缺口连接起来，消除小洞，填补轮廓上的缺口。集合 B 被结构元素 S 断开，记为 $B \circ S$，定义为

$$B \circ S = (B \ominus S) \oplus S \tag{L9.4-1}$$

亦即 B 被 S 断开就是先被 S 腐蚀之后再被 S 膨胀。集合 B 被结构元素 S 闭合，记为 $B \bullet S$，定义为

$$B \bullet S = (B \oplus S) \ominus S \tag{L9.4-2}$$

亦即 B 被 S 闭合就是先被 S 膨胀之后再被 S 腐蚀。

程序范例

1. 目的

了解断开与闭合对图像的实际效应。

2. 程序及说明

```
% 程序 L9_4.m: 断开与闭合 %
% 断开 %
B=imread('L9_4a.bmp');        % 读取图像
S=ones(3,3);                  % 设 S 为一个 3×3 的结构元素
c=erode(B,S,2);               % 将图像用 S 腐蚀两次
d=dilate(c,S,2);              % 将图像用 S 膨胀两次
imshow(B)                     % 显示出原图像
figure,imshow(c)              % 显示出将原图腐蚀两次的图像
figure,imshow(d)              % 显示最后结果图像
 % 闭合 %
B=imread('L9_4b.bmp');        % 读取图像
S=ones(3,3);                  % 设 S 为一个 3×3 的结构元素
c=dilate(B,S,2);              % 将图像用 S 膨胀两次
d=erode(c,S,2);               % 将图像用 S 腐蚀两次
figure,imshow(B)              % 显示原图像
figure,imshow(c)              % 显示将原图像膨胀两次的图像
figure,imshow(d)              % 显示最后处理完毕的图像
```

3. 结果展示



(a) 原图像

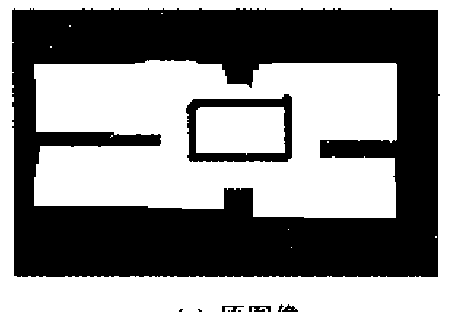


(b) 先腐蚀两次的结果



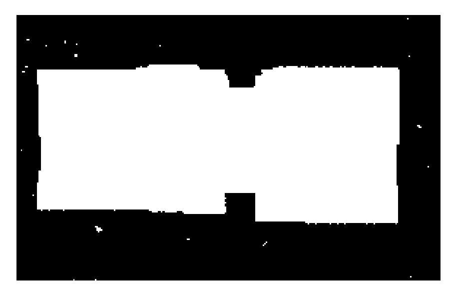
(c) 断开的结果

图　L9.4-1



(a) 原图像

(b) 膨胀之后的结果



(c) 闭合完成后的图像

图 L9.4-2

讨 论

由图 L9.4-1 与 L9.4-2 可以看出断开与闭合对图像的处理效果。值得注意的是，当我们用断开来处理图像的时候，有可能将我们不想去除的较细微处消除、反之闭合也是。所以当我们要用断开与闭合来处理图像的时候，要慎重选择所要处理的图像。

动手做

1. 在闭合程序中，改为先膨胀五次再腐蚀五次，并比较其结果。
2. 在断开程序中，改用 5×5 的结构元素 S 去腐蚀膨胀，并观察其结果。

第十章 实验——图像模式识别

L10.1 利用不变矩判定图形类别

原 理

一个平面图形的几何特性，如大小、位置、方向与形状等，有一些特性与矩有关，其中经由二阶矩与三阶矩可以导出一组七个不变矩，这组不变矩对于平移、旋转及大小的变化不敏感。虽然计算上的误差会造成少许差异，但一般而言不变的特性相当明显。以三类基本简单的几何图形 (已知明显分类) 为训练资料，对每一类图形计算不同 (大小、位置与旋转角度) 所得的七个不变矩，分别计算各类别的平均向量 $\boldsymbol{m}_j$ 及协方差矩阵 $\boldsymbol{C}_j$，作为各类图形的特征，输入不同 (待判别) 的图形，以高斯图像模式的贝氏分类器判定出此图形的正确类别。整个方法描述如下。

$$D_j(\boldsymbol{x}) = -\frac{1}{2}\ln\left|\boldsymbol{C}_j\right| - \frac{1}{2}\left[(\boldsymbol{x}-\boldsymbol{m}_j)^{\mathrm{T}}\boldsymbol{C}_j^{-1}(\boldsymbol{x}-\boldsymbol{m}_j)\right] \tag{L10.1-1}$$

将待判别图形的七个不变矩向量 $\boldsymbol{x}=[\phi_i]_{i=1,2,\cdots,7}$ 输入(L10.1-1)式计算 $D_j(\boldsymbol{x})$ 值，取最大 $D_j(\boldsymbol{x})$ 值，则该图形即可判定成第 j 类。

执行步骤：

(1) 输入训练集图形，计算 ϕ_i，再计算出各类别的 $\boldsymbol{m}_j$ 及 $\boldsymbol{C}_j$。

(2) 输入待判定图形，分别计算 $D_j(\boldsymbol{x})$。

(3) 比较出 $D_j(\boldsymbol{x})$ 的大小，设找出最大者为 $D_k(\boldsymbol{x})$，则此图形判定为第 k 类。

程序范例

1. 目的

计算出三幅图形的七个不变矩，并利用这些不变矩判断出新输入图形的类别。

2. 程序及说明

```
% 程序 L10_1.m: 利用不变矩判定图形类别 %


% 计算圆形的不变矩 %
[X,map]=bmpread('L10_1.bmp'); % invariancem.m 为 L9.1 的程序
C(1,:)=invariancem(X,0.1,0);      % 将图形变为原先的 0.1 倍，并旋转 0 度
```

```
C(2,:)=invariancem(X,0.5,30);       % 将图形变为原先的 0.5 倍，并旋转 30°
C(3,:)=invariancem(X,1,180);        % 将图形变为原先的 1 倍，并旋转 180°
C(4,:)=invariancem(X,1.2,200);      % 将图形变为原先的 1.2 倍，并旋转 200°
C(5,:)=invariancem(X,0.8,300);      % 将图形变为原先的 0.8 倍，并旋转 300°
C_mean=mean(C);                     % 计算五组不变矩的平均值
C_var=cov(C);                       % 计算五组不变矩的协方差

% 计算 X 形的不变矩 %
[X,map]=bmpread('L10_1b.bmp');
x(1,:)=invariancem(X,0.1,0);
x(2,:)=invariancem(X,0.5,30);
x(3,:)=invariancem(X,1,180);
x(4,:)=invariancem(X,1.2,200);
x(5,:)=invariancem(X,0.8,300);
x_mean=mean(x);
x_var=cov(x);

% 计算椭圆形的不变矩 %
[X,map]=bmpread('L10_1c.bmp');
E(1,:)=invariancem(X,0.1,0);
E(2,:)=invariancem(X,0.5,30);
E(3,:)=invariancem(X,1,180);
E(4,:)=invariancem(X,1.2,200);
E(5,:)=invariancem(X,0.8,300);
E_mean=mean(E);
E_var=cov(E);

% 输入一个待测图形，并判断其类别 %
[X,map]=bmpread('L10_1a.bmp');          % 圆形
% [X,map]=bmpread('L10_1b.bmp');        % X 形
% [X,map]=bmpread('L10_1c.bmp');        % 椭圆形

t1=invariancem(X,0.7,0);                % 找出这个待测图形的七个不变矩
D(1,:)=(-1/2)*logm(det(C_var))-(1/2)*(t1-C_mean)*inv(C_var)*(t1-C_mean)';
```

```
D(2,:)=(-1/2)*logm(det(x_var))-(1/2)*(t1-x_mean)*inv(x_var)*(t1-x_mean)';
D(3,:)=(-1/2)*logm(det(E_var))-(1/2)*(t1-E_mean)*inv(E_var)*(t1-E_mean)';

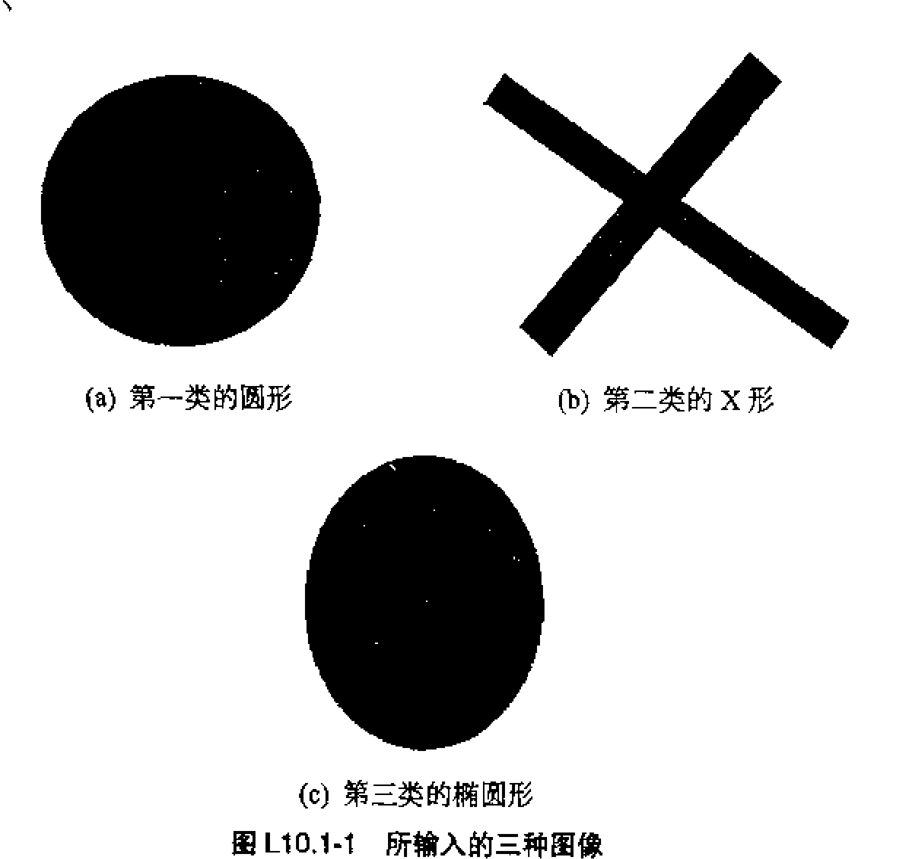
% 找出数值最大者，即判断为此类别 %
maxD=find(D==max(D));

if maxD==1
    disp('This pattern belongs to circle type')
end

if maxD==2
    disp('This pattern belongs to X type')
end

if maxD==3
    disp('This pattern belongs to ellipse type')
end
```

3. 结果展示



(a) 第一类的圆形　(b) 第二类的 X 形

(c) 第三类的椭圆形

图 L10.1-1　所输入的三种图像

讨 论

由执行程序的结果可以知道，椭圆形有时会被误判为圆形，或圆形有时候会被误判为椭圆形，但X形却不会被误判。若观察这三种类别的不变矩便可以发现，圆形及椭圆形的不变矩之间的差异不大，但两者均和X形的不变矩有极大差异。由此可知图形在视觉上的差异越大，其不变矩的差异也就越大，相对的也就越容易被正确无误地判断出其类别。

由上述讨论可以知道，此不变矩的方法较适合应用于样本之间有一定差异性的判读。

动手做

1. 请利用小画家制作形状异于本实验的图形，如梯形、正方形等，计算出其七个不变矩，并观察与实验中三种形状的不变矩的差异。

2. 利用上题1的结果尝试运用实验中的判断式，观察其较容易被归类为哪一类，是否和直观的判断相同。

L10.2 模糊聚类

原 理

模糊聚类的基本原理是在给定的样本中找出各样本与各群的归属度矩阵。第 i 个样本 $\boldsymbol{x}_i$ 若属于第 j 类，则其必有较大的归属值 u_{ij}。详细原理请看本书10.3-3节。

程序范例

1. 目的

随机产生一些2维的数据，并依照10.3-3节的步骤实现FCM算法。

2. 程序及说明

```
% 程序 L10_2.m: 模糊聚类 %
% 模糊 C-平均 (FCM) 算法 %
% N:输入样本数, dimen:样本维度 %
% X:输入样本 (N*dimen) %
% cluster:群数, r:指数型权重(1<r) %
% tolerance:误差临界值 %
% U:分割矩阵(大小为 cluster*N) %
```

```
clear; figure;
X= rand(200,2);
[N,dimen]=size(X);
plot(X(:,1),X(:,2),'*'), axis([-0.1 1.1 -0.1 1.1])
hold on

% 步骤一: 设定一些初值及初始分割矩阵 %
cluster=4; r=2; tolerance=1e-5; cycle=500;
U=rand(cluster,N);
ss=sum(U);
U_new=U./ss(ones(cluster,1),:);                    % 初始模糊分割矩阵

for L=1:cycle
% 步骤二: 计算模糊聚类中心 %
  U=U_new;
  Uexp=U.^r;
  C=Uexp*X./( (ones(dimen,1)*sum(Uexp'))' ); % 新群心的值
  plot(C(:,1),C(:,2),'ro')

% 步骤三: 计算新的分割矩阵 %
  for k=1:cluster
    dist(k,:)=sqrt(sum( ((X- ones(N,1)*C(k,:) ).^2)' ));
  end
  temp=dist.^(-2/(r-1));                          % 计算新的分割矩阵 (1<r)
  U_new=temp./(ones(cluster,1)*sum(temp));

% 步骤四: 计算成本函数值 %
  cost(L)=sum(sum((dist.^2).*Uexp));              % 成本函数
  fprintf('递归循环次数 = % d, 成本函数值 = % f\n', L, cost(L));
  if L>1,
    if abs(cost(L)-cost(L-1))<tolerance
      break;
    end,
  end
end
hold off
disp('最后群心的值 ='); disp(C)
```

3. 结果展示

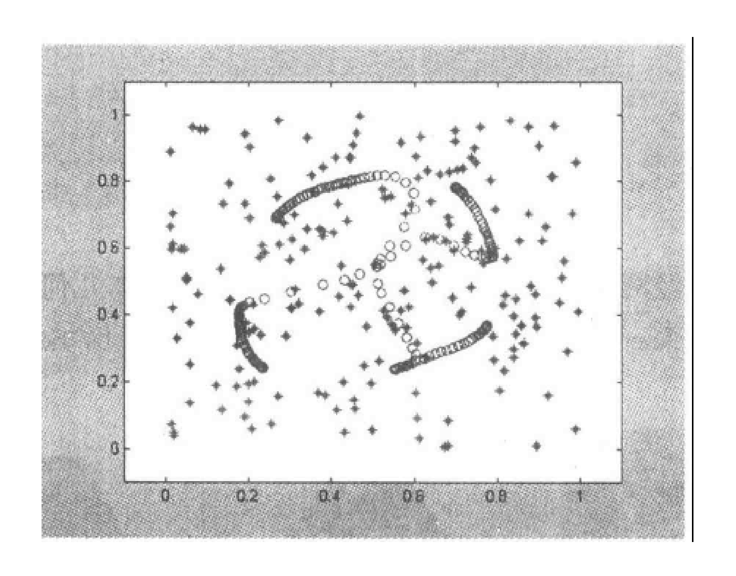


图 L10.2-1 将输入样本分为四类时，执行 FCM 算法的聚类结果

递归循环次数 = 56，成本函数值 = 4.901436
最后群心的值 =

0.699 3	0.783 2
0.234 9	0.241 2
0.776 1	0.366 4
0.265 9	0.689 0

讨 论

整个过程应用了模糊理论中分割矩阵的方法求出群心坐标，此分割矩阵即代表了各样本在各群的归属值。接下来要做的工作可以很直觉的知道，对一新图像模式计算其与各群心的距离，如此即可做一简单分类。

动手做

1. 改变指数型权重 r 的值，观察其在迭代过程中对群心位置变化的影响。

L10.3 利用人工神经网络做图像模式识别

原 理

人工神经网络是一种非线性的映射方式，它将输入的特征值映射到网络的输出分类结果，并可依照其分类的误差大小或某些能量函数来调整网路中的加权值使其达到收敛。以下

我们使用被广为应用的反向传播网络来说明其在分类辨识上的应用。有关反向传播网络的介绍请看本书 10.4-3 节。

程序范例

1. 目的

以反向传播人工神经网络辨识 L 形、三角形及梯形三个基本几何图形，不论其被放大、缩小或旋转均能正确归类。这三种图形的基本样式如图 L10.3-1 所示。

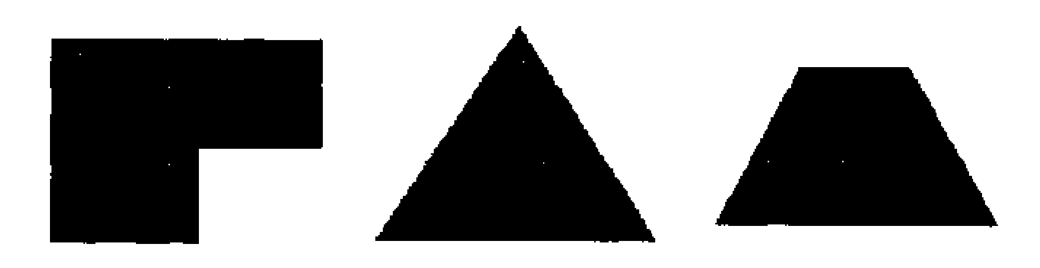


图 L10.3-1 三个待辨识的基本几何图形

利用 L9.1 的程序，分别对 L 形、三角形及梯形三张图像做缩小及放大。在将原图、放大版及其缩小版均旋转 0°、90°、180°、270°，如此每张图像可得到 12 组特征，三张图像共 36 组特征，分别列于表 L10.3-1 至表 L10.3-3。

表 L10.3-1 L 形的特征

缩 小	0.879 0	2.862 0	5.224 7	4.692 5	10.339 3	6.237 3	9.666 4
	0.879 3	2.856 0	5.360 0	4.691 2	10.351 1	6.215 2	9.317 5
	0.879 1	2.845 8	5.484 4	4.696 6	10.328 7	6.213 0	9.821 2
	0.878 7	2.848 0	5.272 0	4.725 7	9.883 7	6.232 9	9.329 4
原 图	0.879 0	2.862 0	5.224 7	4.692 5	10.339 3	6.237 3	9.666 4
	0.879 1	2.858 8	5.294 4	4.691 8	10.780 8	6.225 5	9.276 5
	0.879 0	2.853 8	5.346 9	4.694 6	10.342 7	6.224 6	9.738 5
	0.878 8	2.855 2	5.258 7	4.708 8	10.186 0	6.234 0	9.281 1
放 大	0.878 9	2.862 0	5.224 7	4.692 5	10.339 3	6.237 3	9.666 4
	0.879 0	2.860 4	5.259 9	4.692 2	11.720 6	6.231 2	9.255 1
	0.879 0	2.857 9	5.284 4	4.693 6	10.344 3	6.230 8	9.701 4
	0.878 9	2.858 5	5.244 3	4.700 5	10.567 2	6.235 3	9.257 1

表 L10.3-2　三角形的特征

缩　小	0.924 9	2.869 3	7.937 0	5.453 8	12.425 3	6.893 9	12.409 1
	0.925 1	2.868 3	6.708 7	5.426 3	11.615 2	6.868 0	11.757 0
	0.924 5	2.858 8	6.276 1	5.427 8	11.418 9	6.875 6	11.553 0
	0.924 6	2.860 4	6.206 6	5.394 7	11.227 2	6.839 1	12.926 9
原　图	0.923 8	2.867 0	8.227 2	5.456 2	12.622 7	6.893 9	12.360 2
	0.923 9	2.866 4	7.213 2	5.441 7	11.945 7	6.880 2	11.929 5
	0.923 7	2.861 9	6.825 6	5.442 9	11.806 3	6.882 8	11.730 8
	0.923 7	2.863 0	6.806 6	5.425 0	11.563 0	6.864 3	12.403 7
放　大	0.923 6	2.868 4	8.721 5	5.471 9	13.599 4	6.907 6	12.575 6
	0.923 6	2.868 0	7.892 6	5.464 4	12.529 8	6.900 3	12.189 9
	0.923 5	2.865 8	7.463 7	5.465 5	12.364 2	6.901 3	11.973 8
	0.923 6	2.866 4	7.546 3	5.456 0	11.988 1	6.891 8	12.443 3

表 L10.3-3　梯形的特征

缩　小	0.917 8	2.937 5	6.512 1	5.873 9	12.679 2	7.368 5	11.928 0
	0.918 0	2.936 6	6.649 3	5.832 7	13.248 0	7.330 1	12.170 8
	0.917 7	2.926 5	6.280 9	5.837 4	11.810 9	7.350 9	11.781 5
	0.918 8	2.929 8	6.944 2	5.760 4	12.219 6	7.266 4	12.770 0
原　图	0.916 3	2.935 9	6.488 2	5.885 0	12.686 2	7.381 5	11.914 4
	0.916 5	2.935 4	6.599 6	5.862 6	12.541 4	7.360 3	12.214 7
	0.916 3	2.930 4	6.475 8	5.865 8	12.205 4	7.371 7	11.829 6
	0.916 8	2.932 2	7.130 1	5.825 3	13.289 7	7.327 0	12.379 5
放　大	0.915 6	2.935 2	6.473 2	5.890 5	12.662 2	7.386 7	11.915 3
	0.915 7	2.935 0	6.537 8	5.878 7	12.378 2	7.375 2	12.246 7
	0.915 6	2.932 4	6.499 9	5.880 8	12.752 3	7.381 4	11.866 6
	0.915 9	2.933 4	6.762 1	5.859 7	12.498 8	7.358 6	12.310 7

给定的目标输出值如下：

第一类 (L 形) 都是 (1,0,0)、第二类 (三角形) 都是 (0,1,0)、第三类 (梯形) 都是 (0,0,1)。

网络的学习速度需随训练循环的增加而减少，此表示当训练愈多时则网络愈接近局部最小值，此时应将学习的步伐减小使其趋近最佳值，以免其在加权空间中震荡。程序中并加入惯性项以加速收敛，所谓惯性项表示上一次修正量的部分值。

2. 程序及说明

```
% 程序 L10_3.m: 利用人工神经网络做图像模式识别 %
% X: 输入训练向量 %
% T: 输出目标向量 %
```

```
% h1: 第一层隐藏层神经元个数 %
% cycle: 训练循环 %
% 函数 bpn1ha()为反向传播网络训练程序 %
clear
load in_fea;
X=fea(:,1:7);
T=fea(:,8:10);
h1=7; cycle=10000;
result=bpn1ha(X,T,h1,cycle);

% 函数 bpn1ha.m: 一层隐藏层反向传播网络 (随机选择训练数据) %
function result=bpn1ha(X,T,h1,cycle)
% X:输入训练向量 %
% T:输出目标向量 %
% h1:第一层隐藏层神经元个数 %
% cycle:训练循环 %
% lrate(eta):学习速度 %
X=X/norm(X);             % 输入向量归一化
[N,x]=size(X);
y=size(T,2);
lrate=0.5;

% 产生初始乱数加权值矩阵 %
tseed=prod(clock); rand('seed',tseed);
Wxh=(rand(x,h1)-0.5);              % 输入层—隐藏层间连结加权值
Why=(rand(h1,y)-0.5);              % 隐藏层—输出层间连结加权值
theta_h1=(rand(1,h1)-0.5);         % 隐藏层神经元阈值
theta_y=(rand(1,y)-0.5);           % 输出层神经元阈值

% 配置一些程序初值及存储空间 %
pWxh=0; pWhy=0; ptheta_h1=0; ptheta_y=0;
Lrms=zeros(1,cycle); eta=Lrms;

for L=1:cycle
  Nrms=0; L
  lrate=lrate*0.999; eta(L)=lrate;         % 学习速度衰减
```

```
  s=1:N;
  for n=1:N
    a=length(s);                          % 随机选择训练数据
    b=fix(((a+0.4)-1)*rand(1)) +1;
    n=s(b);   s(b)=[];

% 计算神经元输出值 %
    net1=X(n,:)*Wxh-theta_h1;
    H1=(exp(net1)-exp(-net1))./(exp(net1)+exp(-net1));
    net2=H1*Why-theta_y;
    Y=(exp(net2)-exp(-net2))./(exp(net2)+exp(-net2));

% 计算均方根误差 %
    rms=sqrt(sum((T(n,:)-Y).^2)/y);
    Nrms=Nrms+rms;

% 计算修正量 %
    Dy=(1+Y).*(1-Y).*(T(n,:)-Y);
    DWhy=eta(L)*(H1'*Dy);
    Dtheta_y=-eta(L)*Dy;
    Dh1=(1+H1).*(1-H1).*(Dy*Why');
    DWxh=eta(L)*(X(n,:)'*Dh1);
    Dtheta_h1=-eta(L)*Dh1;

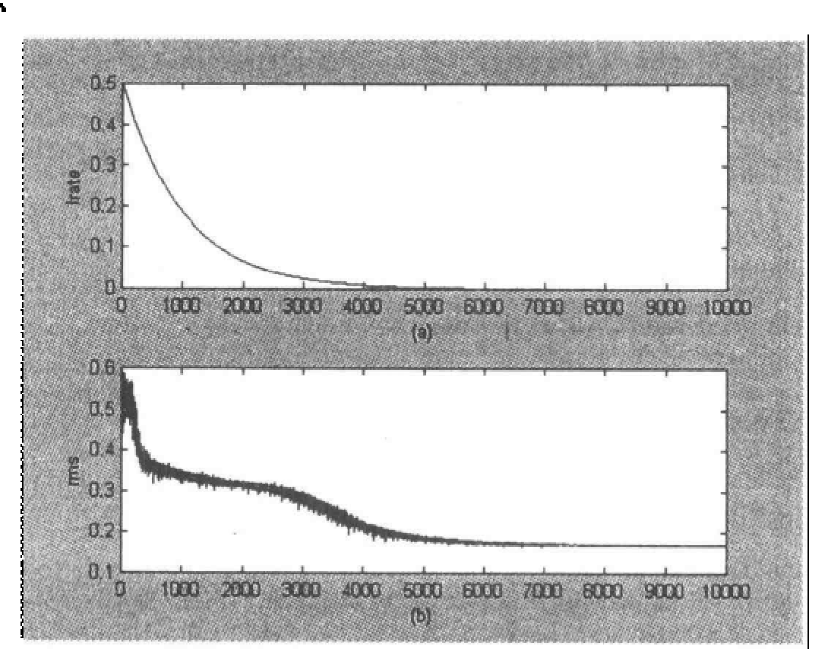
% 更新加权值及阈值 %
    Why = Why + Dwhy + pWhy;
    Wxh = Wxh + DWxh + pWxh;
    theta_y  = theta_y  + Dtheta_y  + ptheta_y  ;
    theta_h1 = theta_h1 + Dtheta_h1 + ptheta_h1;

    pWxh = rms*DWxh;                          % 惯性项
    pWhy = rms*Dwhy;
    ptheta_h1 = rms*Dtheta_h1;
    ptheta_y  = rms*Dtheta_y;
  end
  Lrms(L) = Nrms/N;
end
```

```
% 结果 %
Final_rms = Lrms(cycle)
subplot(211)，plot(eta)，ylabel('lrate')，xlabel('(a) ')
subplot(212)；plot(Lrms)，ylabel('rms')，xlabel('(b)')

% 反向过程 (inside test) %
result = []；corr = 0；
for n = 1：N
   net1 = X(n, : )*Wxh - theta_h1；
   H1 = (exp(net1) - exp( - net1))./(exp(net1) + exp( - net1))；
   net2 = H1*Why - theta_y；
   Y = (exp(net2) - exp( - net2))./(exp(net2) + exp( - net2))；
   [val, index] = max(Y)；     Y = zeros(1, y)；
   Y(index) = 1；
   result = [result；Y]；
   if Y = = T(n, : )
       corr = corr + 1；
   end
end
corr              % 分类正确个数
(corr/N)*100      % 识别率
```

3. 结果展示



注：(a) 学习速率；(b) 均方根误差

图 L10-3.2　反向传播网络参数的变化情形

```
Final_rms =        (最后的均方根误差值)
    0.1640
corr =             (正确分类的样本个数)
    36
ans =              (识别率)
   100
result =           (测试各个样本的结果)
```

36 组向量中第 1 ~ 12 个的输出均为 (1,0,0)，表示其被判断为 L 形；第 13 ~ 24 个的输出均为 (0,1,0)，表示其被判断为三角形；第 25 ~ 36 个的输出均为 (0,1,0)，表示其被判断为梯形。

讨　论

三角形与梯形在视觉上是可明显区分的两类，但从特征值上来看，两者几乎无明显差距。

实验中选择不同的学习速度衰减量、不同的隐藏层神经元个数及不同的训练循环对结果均有影响，其中学习速度衰减量最敏感。经过多次尝试，当选择 0.999 时可有良好收敛，太大或太小的值均无法得到最好的结果。

使用反向传播网络对于此种无明显差距的特征向量依然能够完成分类，由此可见其在分类辨识上的潜力。

动手做

1. 改变隐藏层神经元 h1 的个数，并观察其是否将影响最后收敛的结果。

L10.4　以反向传播网络做模糊分类

原　理

模糊理论与神经网络的结合是一种相当新的应用方式，其中一种结合方式的基本概念是先以模糊化的方式对输入特征做预处理，或是经由各样本间的关系计算出其各自的归属程度，再将这些新的数据加入神经网络中或替代网络中的某些连结，以加速网络的收敛速度或降低网络的复杂度。

程序范例

1. 目的

使用模糊化的方法先对输入数据做预处理再进行识别，观察其对网络收敛的影响。利用

实验 L10-3 中得到的 36 组不变矩特征进行三个基本图像模式的识别, 不同之处在于目标输出值的给定。此处是将经过模糊化预处理后得到的归属值代替目标输出。于是目标输出值不再是 0 或 1，而是界于 0 ~ 1 之间的值，所以网络的反向过程也必须稍加修改。

2. 程序及说明

```
% 程序 L10_4.m: 以反向传播网络做模糊分类 %
clear;
% 模糊化预处理%
C=3;                                    % 分类个数
arfa=2; beta=1.25;
load fea
X=fea; N=size(X,1);
c1=X(1:12,:); c2=X(13:24,:); c3=X(25:36,:);
M=[mean(c1); mean(c2); mean(c3)];  % 各类的平均值
V=[ std(c1);   std(c2);   std(c3)];     % 各类的标准差
for i=1:N                               % 加权距离矩阵
    Z(i,:)=sqrt(sum((((ones(C,1)*X(i,:)-M)./V).^2)'));
end
meu=1./(1+(Z/arfa).^beta);              % 各样本的归属值
%以反向传播网络进行识别%
T=meu; h1=7; cycle=10 000;              % 将归属值当成目标输出
result=bpn1hb(X,T,h1,cycle);            % bpn1hb.m 与 L10_3.m 中的 bpn1ha.m
                                        % 在反向过程稍有不同, 请自行比较
% 函数 bpn1hb.m: 一层隐藏层反向传播网络 (随机选择训练数据) %
% X:输入训练向量 %
% T:输出目标向量 %
% h1:第一层隐藏层神经元个数 %
% cycle:训练循环 %
% lrate(eta):学习速度 %

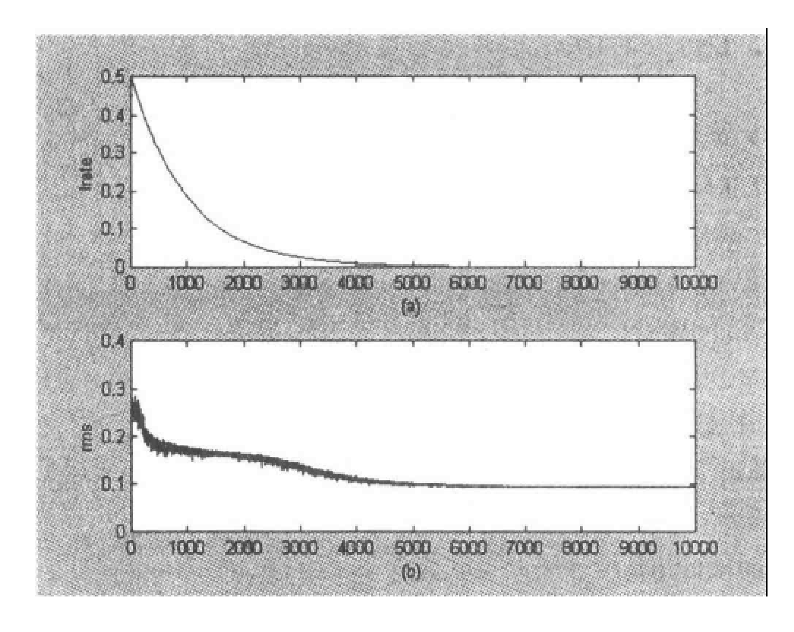
function result =bpn1hb(X,T,h1,cycle)
X=X/norm(X);                            % 输入向量归一化
[N,x]=size(X);
y=size(T,2);
lrate=0.5;
% 产生初始乱数加权值矩阵 %
tseed=prod(clock); rand('seed',tseed);
Wxh=(rand(x,h1)-0.5);                   % 输入层—隐藏层间连结加权值
Why=(rand(h1,y)-0.5);                   % 隐藏层—输出层间连结加权值
```

```
theta_h1=(rand(1,h1)-0.5);        % 隐藏层神经元阈值
theta_y=(rand(1,y)-0.5);          % 输出层神经元阈值
% 配置一些程序初值及储存空间 %
pWxh=0; pWhy=0; ptheta_h1=0; ptheta_y=0;
Lrms=zeros(1,cycle); eta=Lrms;
for L=1:cycle
  Nrms=0; L
  lrate=lrate*0.999; eta(L)=lrate;          % 学习速度衰减
  s=1:N;
  for n=1:N
    a=length(s);                             % 随机选择训练数据
    b=fix(((a+0.4)-1)*rand(1)) +1;
    n=s(b);   s(b)=[];
    % 计算神经元输出值 %
    net1=X(n,:)*Wxh-theta_h1;
    H1=(exp(net1)-exp(-net1))./(exp(net1)+exp(-net1));
    net2=H1*Why-theta_y;
    Y=(exp(net2)-exp(-net2))./(exp(net2)+exp(-net2));
    % 计算均方根误差 %
    rms=sqrt(sum((T(n,:)-Y).^2)/y);
    Nrms=Nrms+rms;
    % 计算修正量 %
    Dy=(1+Y).*(1-Y).*(T(n,:)-Y);
    DWhy=eta(L)*(H1'*Dy);
    Dtheta_y=-eta(L)*Dy;
    Dh1=(1+H1).*(1-H1).*(Dy*Why');
    DWxh=eta(L)*(X(n,:)'*Dh1);
    Dtheta_h1=-eta(L)*Dh1;
    % 更新加权值及阈值 %
    Why=Why+DWhy+pWhy;
    Wxh=Wxh+DWxh+pWxh;
    theta_y =theta_y +Dtheta_y +ptheta_y ;
    theta_h1=theta_h1+Dtheta_h1+ptheta_h1;
    pWxh=rms*DWxh;                          % 惯性项
    pWhy=rms*DWhy;
    ptheta_h1=rms*Dtheta_h1;
    ptheta_y =rms*Dtheta_y;
  end
  Lrms(L)=Nrms/N;
```

```
end

% 结果 %
Final_rms=Lrms(cycle)
subplot(211), plot(eta), ylabel('lrate'), xlabel('(a)')
subplot(212), plot(Lrms), ylabel('rms'), xlabel('(b)')
% 反向过程 (inside test) %
result=[]; corr=0;
for n=1:N
   net1=X(n,:)*Wxh-theta_h1;
   H1=(exp(net1)-exp(-net1))./(exp(net1)+exp(-net1));
   net2=H1*Why-theta_y;
   Y=(exp(net2)-exp(-net2))./(exp(net2)+exp(-net2));
   [val_y,index_y]=max(Y);
   [val_t,index_t]=max(T(n,:));
   result=[result; Y];
   if index_y==index_t
       corr=corr+1;
   end
end
corr              % 分类正确个数
(corr/N)*100      % 识别率
```

3. 结果展示



注：(a) 学习速度；(b) 均方根误差

图 L10.4-1　以反向传播网络做模糊分类的结果

```
Final_rms =        (最后的均方根误差值)
    0.0896
corr =             (正确分类的样本个数)
    36
ans =              (识别率)
   100
```

讨 论

比较图 L10.3 - 2 与图 L10.4 - 1，可看出直接做反向传播网络的实验，大约在第 7 500 个循环左右时网络即不再震荡，而先经模糊化预处理的实验则在第 6 500 个循环附近时即已趋于平稳，此比较说明了做模糊化预处理的好处。

动手做

1. 试着微调 α (arfa)与 β (beta)的值，观察其对归属值 μ 的影响。

附 录

光盘内容简介及使用说明

1. 光盘内容

本说明置于 readme.txt 文件中。在本书第二部分所有的程序范例及相关的图像文件均放在本光盘中。以章为单元分类，每章有独立的子目录，子目录下为各章的内容。以下列出所有文件的名称及其功能。

bmp (额外可供实验的 bmp 图像文件)

ball1.bmp ball2.bmp ball3.bmp ball4.bmp ball5.bmp ball6.bmp
ball7.bmp ball8.bmp ball9.bmp ball10.bmp
cat.bmp
cross1.bmp cross2.bmp
disk1.bmp disk2.bmp disk3.bmp disk4.bmp disk5.bmp disk6.bmp
disk7.bmp disk8.bmp disk9.bmp
gift.bmp
sign1.bmp sign2.bmp

ch2

L2_1.m: 二维卷积 (程序)
L2_2.m: 矩阵的直积 (程序)
L2_3.m: 马尔可夫链的转移概率 (程序)

ch3

L3_1.m: 纯量量化器的设计 (程序)
L3_2.m: 量化造成的假轮廓 (程序)
L3_3.m: 向量量化器的码本的产生 (程序)
L3_4.m: 利用 LBG 训练三个不同大小与维度的码本并分别进行 VQ (程序)
gau.m: ML 量化器设计中分母的计算式 (函数)
gau1.m: ML 量化器设计中分子的计算式 (函数)
LBG.m: LBG 训练法 (函数)
quantize.m: 高斯概率密度函数的非均匀量化 (函数)

VQ.m: 向量量化(函数)
L3_2.bmp: 图像文件
lena.mat: Matlab 的矩阵变量文件

ch4\

L4_1.m: SVD 变换 (程序)
L4_2.m: 图像变换的能量集中能力 (程序)
L4_3a.m: 小波变换的多重解析结构 (程序)
L4_3b.m: 小波变换系数与图像的方向性 (程序)
energy.m: 能量集中 (函数)
L4_1.bmp: 图像文件
L4_2.bmp: 图像文件
L4_3b.bmp: 图像文件
woman.mat: Matlab 的矩阵变量文件

ch5\

L5_1.m: 直方图均衡化法(程序)
L5_2.m: 平滑滤波器(程序)
L5_3.m: 同态滤波器(程序)
L5_1.bmp: 图像文件
L5_2.bmp: 图像文件
L5_3.bmp: 图像文件

ch6\

L6_1.m: 最小平方滤波器 (程序)
L6_2.m: 迭代盲目去卷积法 (程序)
L6_1.bmp: 图像文件
L6_2a.bmp: 图像文件
L6_2b.bmp: 图像文件

ch7\

L7_1.m: 游程长度编码 (程序)
L7_2.m: 应用小波变换与向量量化做图像压缩 (程序)
RLE.m: 游程长度编码 (函数)
LBG.m: LBG 训练法 (函数)
VQ.m: 向量量化 (函数)
L7_1a.bmp: 图像文件
L7_1b.bmp: 图像文件

lena.mat: Matlab 的矩阵变量文件

ch8\

L8_1.m: 像素聚类区域成长法 (程序)
L8_2.m: 四叉树区域分割与合并法 (程序)
L8_3.m: Sobel 边缘检测 (程序)
L8_4.m: 拉氏边界检测法 (程序)
L8_5.m: LUM 滤波器 (程序)
LUM.m: LUM 滤波器 (函数)
regiongrow.m: 像素聚积成长法 (函数)
sobel.m: sobel 算子 (函数)
L8_1.bmp: 图像文件
L8_2.bmp: 图像文件
L8_4.bmp: 图像文件
lena.mat: Matlab 的矩阵变量文件

ch9\

L9_1.m: 不变矩 (程序)
L9_2.m: 细化 (程序)
L9_3.m: 膨胀与腐蚀 (程序)
L9_4.m: 断开与闭合 (程序)
invariance.m: 不变矩 (函数)
thin.m: 细化 (函数)
L9_1.bmp: 图像文件
L9_2.bmp: 图像文件
L9_3.bmp: 图像文件
L9_4a.bmp: 图像文件
L9_4b.bmp: 图像文件

ch10\

L10_1.m: 利用不变矩判定图形类别 (程序)
L10_2.m: 模糊聚类 (程序)
L10_3.m: 利用人工神经网络做图像模式识别 (程序)
L10_4.m: 以反向传播网络做模糊分类 (程序)
bpn1ha.m: 一层隐藏层反向传播网络 (随机选择训练数据) (函数)
bpn1hb.m: 一层隐藏层反向传播网络 (随机选择训练数据) (函数)
invariance.m: 不变矩(函数)
L10_1a.bmp: 图像文件

L10_1b.bmp: 图像文件
L10_1c.bmp: 图像文件
fea.mat: Matlab 的矩阵变量文件
in_fea.mat: Matlab 的矩阵变量文件

2. 使用说明

文件内容分成四类，第一类是一般的程序，第二类是可由程序调用的函数，第三类是供实验用的图像文件，第四类是 Matlab 的矩阵变量文件。所有程序或函数均在 PC 版 Matlab 5.1 的环境下开发完成，其中许多程序和函数会用到 Matlab 的图像工具箱 (image toolbox)，少部分会用到 Matlab 的小波工具箱 (wavelet toolbox)。使用上述程序只要按照 Matlab 的一般规定将相关文件案 copy 到 Matlab 里的适当目录下就可使用了。

参考文献

[1] Arerbuch A., Lazar D., and Israeli M. [1996]. "Image Compression Using Wavelet Transform and Multiresolution Decomposition." *IEEE Trans. Image Processing*, vol. 5, no. 1, pp. 4-15

[2] Ayers G. R. and Dainty J. C. [1988]. "Iterative Blind Deconvolution Method and Its Applications." *Optics Letters*, vol. 13, no. 7, pp. 547-549

[3] Bezdek J. C. [1981]. *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, New York

[4] Carlucci L. [1972]. "A Formal System for Texture Language." *Pattern Recognition*, vol. 4, pp. 53-72

[5] Castelman K. R. [1996]. *Digital Image Processing*, Prentice-Hall

[6] Dudani S. A., Breeding, K. J., and McGhee, R. B. [1977]. "Aircraft Identification by Moment Invariants." *IEEE Trans. Computers*, vol. 26, pp. 39-45

[7] Faber T. L. and Stokely, E. M. [1988]. "Orientation of 3D Structures in Medical Images." *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 10, pp. 626-633

[8] Faugeras O. D. and Partt, W. K. [1980]. "Decorrelation Methods of Texture Feature Extraction." *IEEE* Trans. *Pattern Anal. Mach. Intell.*, vol. 2, pp. 323-332

[9] Forgy E. W. [1965]. "Cluster Analysis of Multivariate Data: Efficiency vs. Interpretability of Classifications. " Abstract, Biometrics, vol. 21, pp. 768-769

[10] Gonzalez R. C. and Wood, R. E. [1992]. *Digital Image Processing*, Addison-Wesley

[11] Gray S. B. [1971]. "Local Properties of Binary Images in Two Dimension." *IEEE Trans. Computers*, vol. 20, no. 5, pp. 551-561

[12] Haralick R. M. and Shapiro, L. G. [1985] "Survey: Image Segmentation." *Comput. Vision, Graphics, Image Processing*, vol. 29, pp. 100-132

[13] Haralick R. M. and Shapiro, L. G. [1993]. *Computer and Robot Vision*, vols. Ⅰ and Ⅱ, Addison-Wesley

[14] Hardie R. C. and Boncelet, C. G. [1993]. "LUM Filters: A Class of Rank-Order Based Filters for Smoothing and Sharpening." *IEEE Trans. Signal Processing*, vol. 41, pp. 1061-1076

[15] Hardie R. C. and Boncelet, C. G. [1995]. "Gradient-Based Edge Detection Using Nonlinear Edge Enhancing Prefilters." *IEEE Trans. Image Processing*, vol. 4, pp.1572-1577

[16] Heijden V. F. [1994]. *Image Based Measurement System -- Object Recognition and Parameter Estimation*, John Wiley & Sons

[17] Hopfield J. J. [1982]. "Neural Networks and Physical System with Emergent Collective Computational Abilities." Proc. *Natl. Acad. Sci. USA*, vol. 79, pp. 2554-2558

[18] Hsia T. C. [1981]. "A Note on Invariant Moments in Image Processing." *IEEE Trans. System, Man, and* Cybernetics, vol. 11, pp. 831-834

[19] Hu M. K. [1962]. "Visual Pattern Recognition by Moment Invariants." *IRE Trans. Info. Theory*, vol. 8, pp. 179-187

[20] Jain A. K. [1989]. *Fundaments of Digital Image Processing*, Prentice-Hall

[21] Kundur D. and Hatzinakos D.[1996]. "Blind Image Deconvolution." *IEEE Signal Processing Magazine*, vol. 13, no. 3, pp. 43-64

[22] Leu J.-G. and Wee W. G. [1985]. "Detecting the Spatial Structure of Natural Textures Based on Shape Analysis." *Computer Vision, Graphics, and Image Processing*, vol. 31, pp. 67-88

[23] Lim J. S. [1990]. *Two-Dimensional Signal and Image Processing*, Prentice-Hall

[24] Lin C-T. and Lee C. S. [1996]. *Neural Fuzzy Systems, -- A Neuro -- Fuzzy Synergism to Iintelligent Systems*, Prentice-Hall, New Jersey

[25] Linde A., Buzo A., and Gray R. M. [1980]. "An Algorithm for Vector Quantizer Design." *IEEE Trans. Commun.*, vol. 28, pp. 84-95

[26] Lu S. Y. and Fu K. S. [1978]. "A Syntactic Approach to Texture Analysis." *Computer Graphics and Image Processing*, vol. 7, pp. 303-330

[27] Pandya S. Abhijit and Macy Robert B. [1996]. *Pattern Recognition with Neural Networks in C++*, CRC Press

[28] Parker J. R. [1997]. *Algorithms for Image Processing and Computer Vision*, John Wiley and Sons

[29] Pennebaker W. B. and Mitchell J. L. [1993]. *JPEG: Still Image Data Compression Standard*, Van Nostrand Reinhold, New York

[30] Porat B. [1997]. *A* Cource *in Digital Signal Processing*, John Wiley and Sons

[31] Pratt W. K. [1991]. *Digital Image Processing*, 2nd ed., John Wiley & Sons

[32] Reeves A. P. and Taylor, R. W. [1989]. "Identification of Three-Dimensional Objects Using Range Information." *IEEE* Trans. *Pattern Anal. Mach. Intell.*, vol. 11, pp. 403-410

[33] Rumelhart D. E., Hinton G. E., and Williams R. J. [1986]. "Learing Internal Representations by Error Propagation." *in Pallel Distributed Processing: Explorations in the Microstructures of Cognition*, vol. 1: Foundations, D. E. Rumelhert and J. L. McClelland (eds.), MIT Press, Cambridge, MA

[34] Schalkoff R. [1992], *Pattern Recognition -- Statistical, Structural, and Neural Approaches*, John Wiley & Sons

[35] The Math Works Inc. [1997]. *Image Processing Toolbox User's Guide, Version 2*

[36] Therrien C. W. [1992]. *Discrete Random Signals and Statistical Signal Processing*, Prentice-Hall, Englewood Cliffs, New Jersey

[37] Tomita F., Shirai Y., and Tsuji S. [1982]. "Description of Texture by a Structural Analysis." *IEEE Trans.* Patttern *Anal. Mach. Intell.*, vol. 4, no. 2, pp. 183-191

[38] Tou J. T. and Gonzalez R. C. [1974]. *Pattern Recognition*, Addison-Wesley

[39] Zucker S. W. [1976a]. "On the Structure of Texture." *Perception*, vol. 5, pp. 419-436

[40] Zuucker S. W. [1976b]. "Toward a Model of Texture." *Computer Graphics and Image Processing*, vol. 5, pp. 190-202

[41] 史学勋. 数位影像压缩.中原大学(台湾省)电子系数位影像处理学期报告，1996

[42] 朱梦杰. 影像分割.中原大学(台湾省)电子系数位影像处理学期报告，1996

[43] 余松煜、周源华，吴时光. 数位影像处理，儒林，1993

[44] 李正男. 盲目影像还原.中原大学(台湾省)电子系数位影像处理学期报告，1996

[45] 林克峰. 多媒体影像及视讯资料的压缩技术. 第三波, pp. 88-93，1998

[46] 施威铭研究室. PC 影像处理技术(一)——图档压缩篇.旗标出版社(台湾省)，1993

[47] 陈永吉. 下一代多媒体技术核心 —— MPEG-2 搜秘. 新电子科技杂志, vol. 113, pp. 141-147，1995

[48] 陈永昌. 结合 LUM 滤波器与类神经网路之加强式边缘检测.中原大学(台湾省)电子系硕士论文，1998

[49] 陈俊旭. 影像增强.中原大学(台湾省)电子系数位影像处理学期报告，1996

[50] 杨武智. 影像处理与辨认. 全华，1994

[51] 叶怡成. 类神经网路模式应用与实作. 儒林，1993

[52] 卫祖赏. 数位影像处理. 全华，1996

[53] 缪绍纲. 工程数学(下册). 修订第二版. 台北："中央图书出版社"(台湾省)，1987

[54] 缪绍纲. 工程数学(上册). 修订第二版. 台北："中央图书出版社"(台湾省)，1987

[55] 缪绍纲. 工程数学习题解答(下册)，台北："中央图书出版社"(台湾省)，1987